Kombilo Documentation

Release 0.7.4

Ulrich Goertz

August 06, 2012

CONTENTS

Kombilo is a go database program. Its main purpose is to search for games in which a given pattern or position occurs. You can also search for other criteria (like time period, players, events).

The name 'Kombilo' is the Esperanto word for comb. It allows you to 'go through the game records with a fine-toothed comb'. Pronunciation: 'i' as 'ee' in see, and the stress is on the 'i'.

Kombilo was written by Ulrich Goertz, ug@geometry.de.

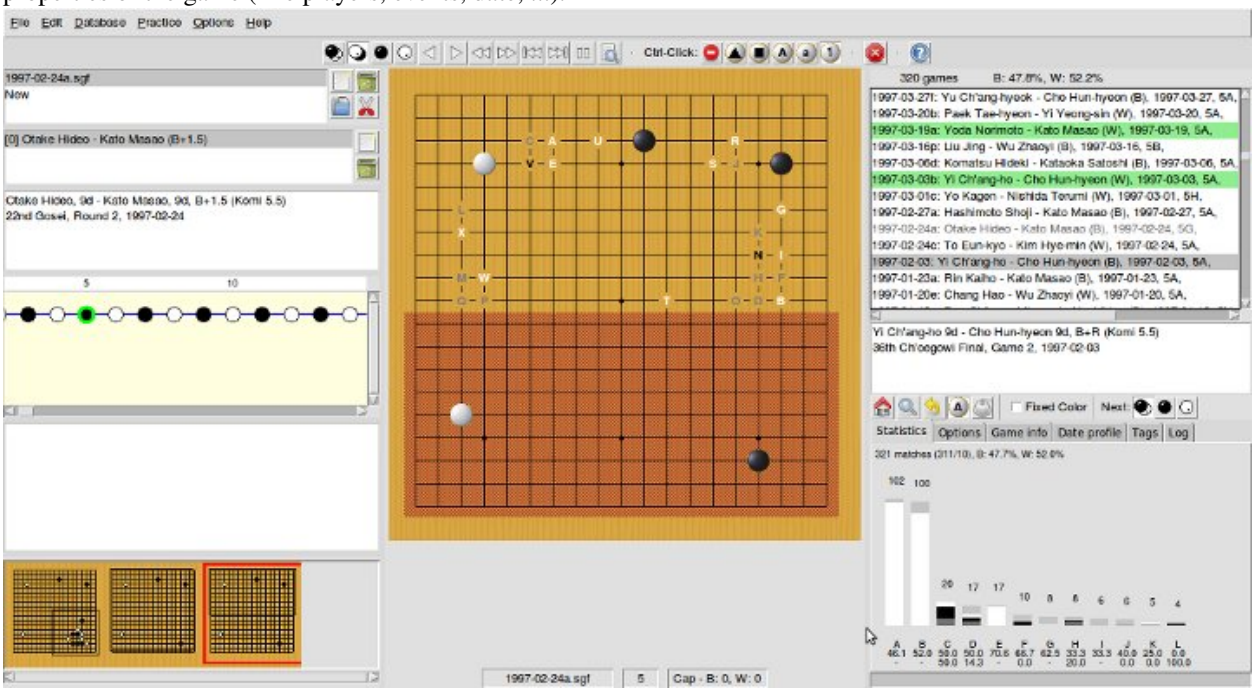
Web site: u-go.net/kombilo/

CHAPTER

ONE

INTRODUCTION

Kombilo is a go database program. Its particular strength is searching for move patterns in a collection of SGF files (like searching for all games where a particular opening or a particular joseki is played). You can also search for other properties of the game (like players, events, date, ...).



1.1 Features

- You can search for full board patterns, or for patterns occurring in a corner, on the side or anywhere on the board. All patterns given by symmetries of the board are found as well, and also - unless you disable it - the pattern obtained by exchanging black and white.
- Kombilo comes with a complete SGF editor: so you can add variations of your own, comment the game, add labels etc. The SGF editor can also handle collections, i.e. SGF files containing several games. The tree structure of the current game is shown in a separate window. You can rotate/mirror SGF files.
- Kombilo has built in list of references to commentaries of games in the English go literature. (NB: Kombilo does not come with the game records, but recognizes the games by the Dyer signature.) Those games in your database which Kombilo finds in its list are marked in the game list, and in the game info a reference to the journal/book

which has the commentary is given. Currently the list contains almost 2000 references, and includes references to the game commentaries in all issues of Go World, and in most English go books with game comentaries.

- You can search for pieces of the *game information*, i.e. for player names, events, date, etc., and you can issue complex queries by directly accessing the underlying SQL database.
- After any combination of searches, you can quickly have a *date profile* of the current list of games displayed.
- You can refine pattern searches in many ways: by fixing who should move next in the search pattern, by allowing or disabling search for the pattern with black/white exchanged, by requiring that the pattern should occur before some specified point in the game (before move 50, say), by searching for move sequences, etc. By default, this version of Kombilo also searches in variations. The pattern search has been "parallelized" and hence can use several processor cores. Depending on your hardware, this results in a significant speed-up in comparison to older Kombilo versions.
- In addition to the graphical user interface, there is an interface for using the underlying functionality of Kombilo within Python scripts, or as a C++ library.
- Kombilo is free, and is open source. Your *contributions* are welcome. Feel free to freely distribute it, and feel free to clone or fork the project on BitBucket. I will be glad to consider patches you send me for inclusion into the Kombilo code.
- Kombilo has been developed with tools that are available on all major operating systems (at least Linux, Windows, Mac OS X), *but the current version has only been tested on (Ubuntu) Linux*. Probably some twists will be required in order to get everything running smoothly on Windows and/or Mac OS X; your help will be much appreciated.
- If you think some feature is missing, or if you found a bug, please open a ticket on BitBucket, or send me an email.

CHAPTER

TWO

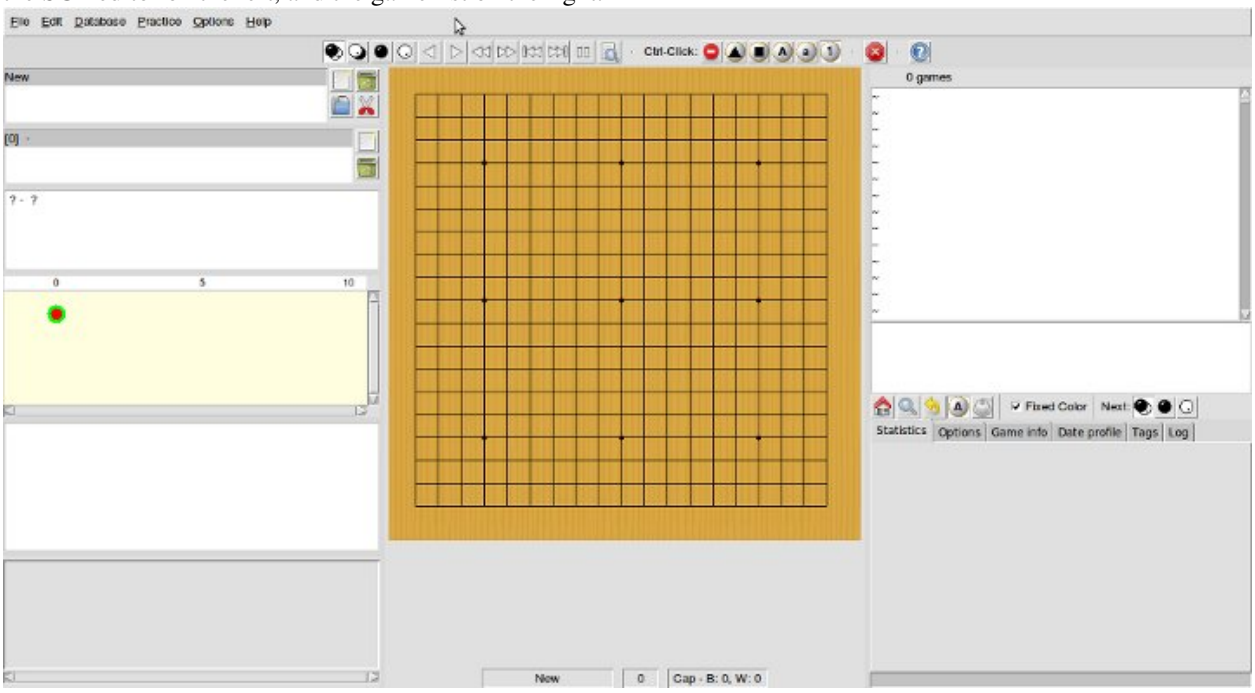
TUTORIAL: GETTING STARTED WITH KOMBILO

This document give you a step-by-step tutorial to setting up Kombilo and to getting started with the program.

2.1 Getting started

First you need to *install* the program. For Windows, there is an installer which will set up everything for you. For other operating systems, or if there are problems, look at the pertaining instructions: *Linux*, *Windows*, *Mac OS X*.

When you start Kombilo, the **main window** will open, with a go board in the middle, some widgets which belong to the SGF editor on the left, and the game list on the right.

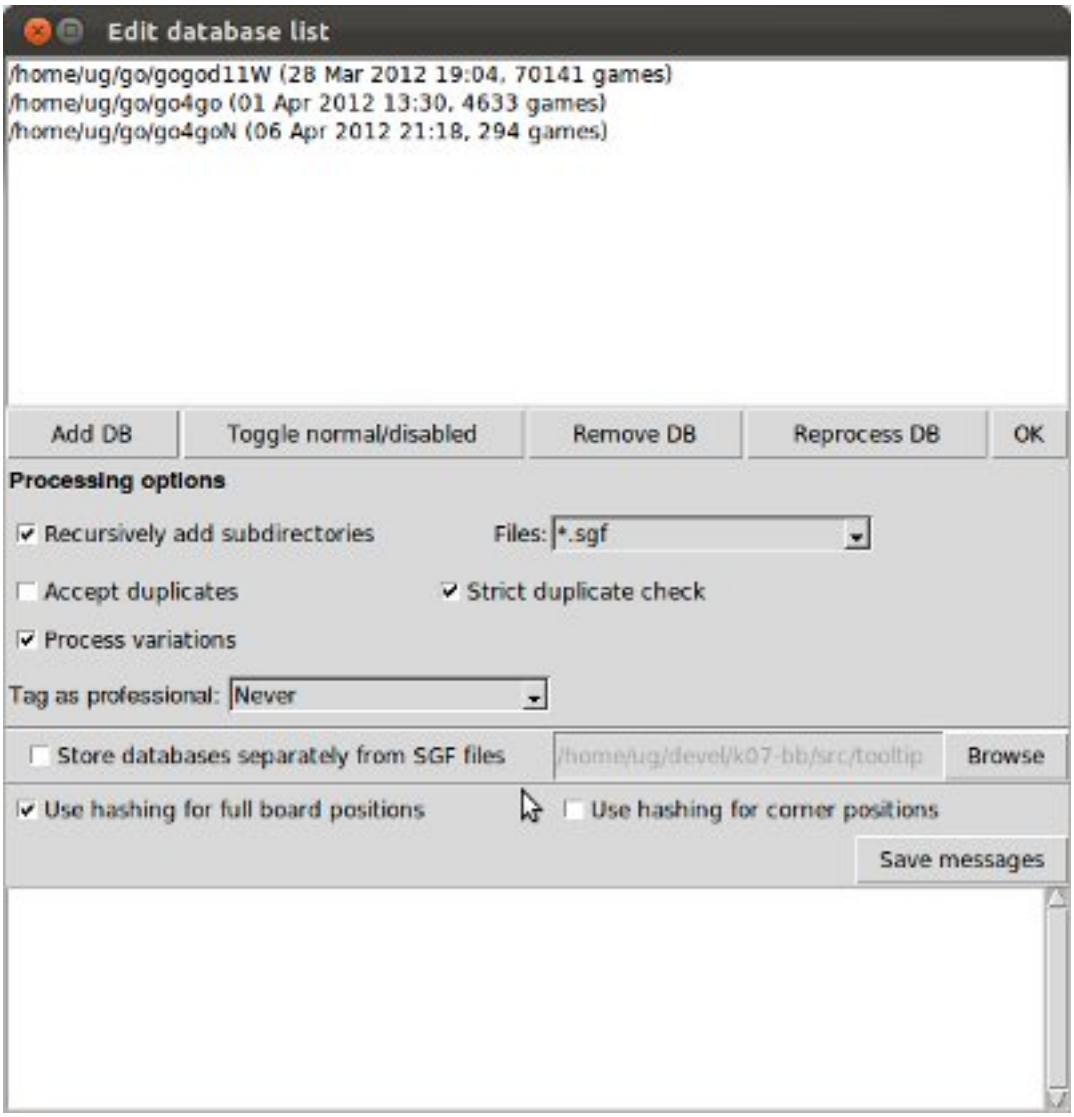


You can change the sizes of the columns by dragging the *sashes* between them, and similarly for the horizontal panes in the left hand column.

The first thing you have to do now is to add SGF files to the database list; choose the *Edit DB list* command in the Database menu.

A *database* corresponds to a directory of SGF files; it contains all the SGF files in that directory. Kombilo does not come with any games. You can either download a game collection (*Where to find game records*) from the internet, or buy a commercial one. The GoGoD encyclopedia comes with more than 70.000 at the time of writing, and is highly recommended.

See also the section about *Requirements on SGF files* in order to understand which kind of SGF files the program can handle.



Check (and change, if necessary) the options - mainly whether all subdirectories should also be added, and then use the `Add DB` button to select a directory of SGF files. You can add several directories one after the other. From a performance point of view, it is best not to have too many databases, but rather to group your SGF files into few databases.

When you add a directory for the first time, the SGF files will be 'translated' into a format that makes the search more efficient. This processing takes quite some time; if you have thousands of games, it will take a couple of minutes even on a very fast machine. But this has only to be done once. The data will be written to several .db files in the same directory. (Note: this processing is much faster now than it was in Kombilo 0.5 and earlier versions.)

The sgf files remain in the directory, and Kombilo will not change them (unless you change the game info or edit the games yourself, of course). After the processing, the pattern search function actually does not use them anymore, but they are needed if you want to play through games with the SGF viewer.

After having added one or more databases, close the `Edit DB list` window using the `OK` button.

> **Warning:** Character encodings
> Currently, this version of Kombilo works well only with **UTF-8** encoded files. Most SGF files produced in the western world are pure ASCII (which is a subset of UTF-8, and in particular can be handled by Kombilo), but many SGF files with asian characters are encoded using different character encodings and hence cannot be fed into Kombilo right away. This issue will hopefully be resolved soon. See *Encodings*.

2.2 Searching for patterns

Now the game list should contain some files, and you can start the first search. Place stones on the board by clicking. Ctrl-click to remove stones, and Shift-click to place *wildcards*.

With the right mouse key (click and drag) you can select the search-relevant region; the rest of the board will be grayed out.

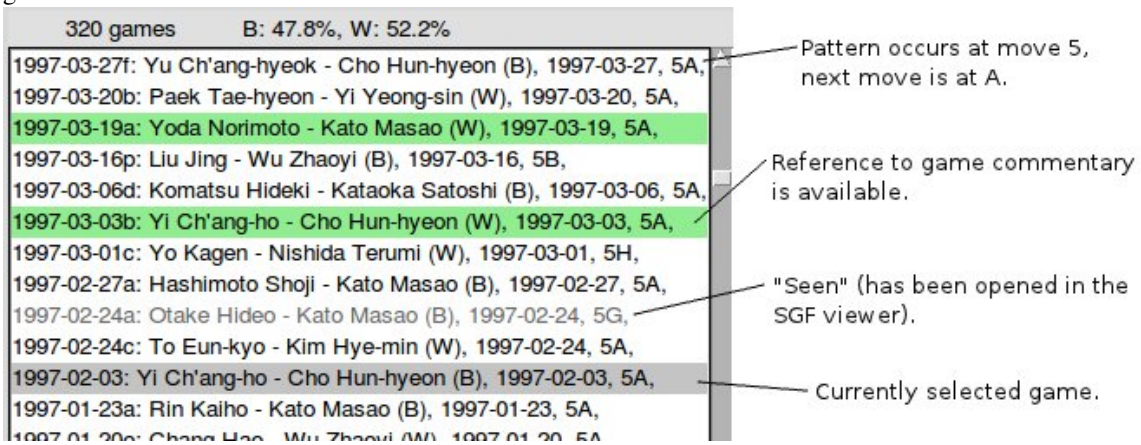
Everything outside that region is ignored: on the one hand it does not matter if there are additional stones on the main board, on the other hand all games will be found which feature the given pattern in the relevant region, no matter what else is on the board. Of course, mirroring and rotating the board is automatically taken into account.

Switch between placing black and white stones alternatingly, or stones of one color only by using the left most buttons in the toolbar.

When no region is selected, the whole board is relevant.

After defining the pattern and the relevant region, just click the search button (the looking glass in the row of buttons directly below the game list), or press `Control-p`. In order to go back to the complete game list, use the "reset game list" button - the leftmost button below the game list, or press `Control-r`.

If you click on a game in the game list, the game info (players, result, komi, event, date etc.) is displayed below the game list.



By double-clicking on a game in the game list, you load the game to the SGF editor and you can look at that game. You can also start the viewer by selecting a game (by a single click) and pressing the return key. If you prefer to open the game in a new window, use Shift-Click instead of double-clicking; cf. the *corresponding option*.

If you prefer, you can use your customary SGF editor instead of the SGF viewer coming with Kombilo; use the 'Alternative SGF viewer' command in the Options menu.

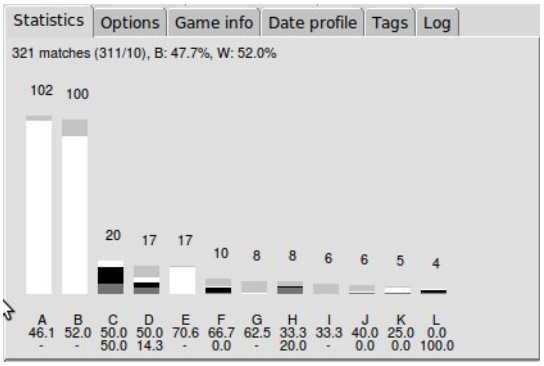
By clicking on a game with the right mouse key, a window will pop up where the complete game info is displayed, and can be edited. **Note:** By pressing OK in this window, the game info is written to the original SGF file.

In the `Statistics` tab in the lower portion of the right hand column, some statistics will be shown about the continuations in the given position.

In the first line you find the number of hits (which, obviously, can be bigger than the number of games in the list); after this number, in parentheses, is the number of matches with colors as on the board respectively reversed colors. Finally, you get the B/W winning percentages corresponding to the hits (i.e. a game where the pattern occurs several times, is counted that often).

Below some information on the continuations in the search position is given. For the ten most frequent continuations, you get

- the number of hits in which this continuation is played
- graphically, it is shown, how often white played at this point after a tenuki (light gray), how often white played there directly after the pattern was finished (white), how often black played there directly after the pattern was finished (black), and finally how often black played after tenuki at the given point.
- finally, below the letter labelling the corresponding point on the board (use the button with the labeled white stone to display the labels on the board), you get the black winning percentage for white playing at this point, and then the black winning percentage for black playing there. (Because there is not enough space, the winning percentage for white is not given, but of course (neglecting jigos etc.) it will be 100% - black winning percentage.



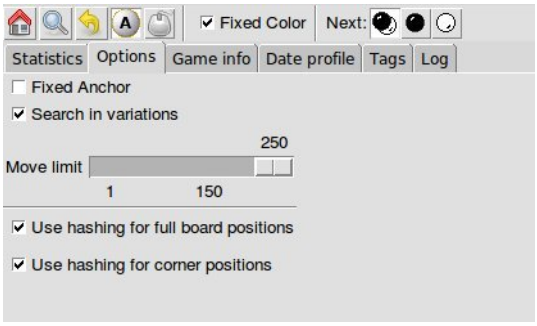
The labels are ordered by the number of occurrences of the corresponding continuation. (Unless there were already labels present in the search pattern: in that case Kombilo will use those labels to refer to the same intersections, and thus will not sort by frequency.)

If you have a sufficient number of games in your databases, this lets you create fuseki and joseki dictionaries very easily: The color of the label indicates whether black or white (or both, depending on the game, in case of the gray labels) played on this point. See `sgftree`.

After a search, you can clear the board with the *start* button above the board. You can reset the game list (such that it contains all the games again) with the *reset game list* button in the toolbar below the game list, or by pressing `Control-r`. In the file menu, you can also choose to do a "complete reset" - that will reset Kombilo to the state right after it started up.

2.3 Pattern search options

There are several buttons to customize the search in the game list window:



Usually the pattern obtained by reversing the colors is searched for too, but you can disable that with the 'fixed color' option.

As a default, Kombilo uses the 'smart fixed color' option, which automatically enables 'fixed color' for whole board searches, and disables it for all other searches. You can change that in the *Options in the Options menu*.

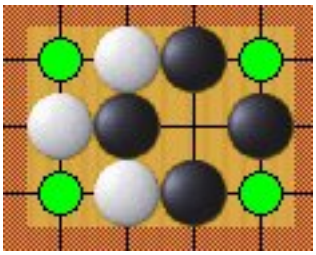
Furthermore, for a pattern on the edge or in the middle of the board, the program also looks for translations; this can be disabled by the *fixed anchor* option.

With the *black/white*, *black* and *white* buttons in the line below you can limit the search to patterns where black plays next or white plays next. This is sometimes useful, in particular for joseki searches with very few stones on the board. The default is to allow either a black or a white continuation (or no continuation at all).

Finally, you can impose a move limit, such that only games are found where the pattern occurs before the given limit.

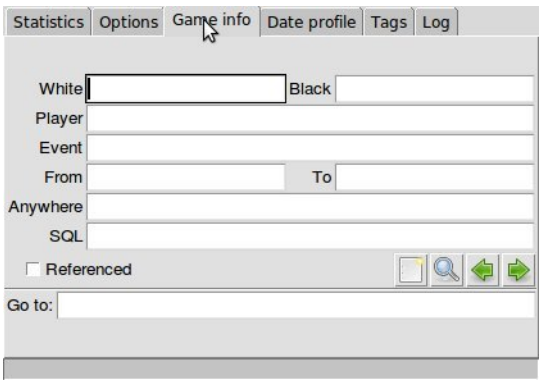
You can also add wildcards to the search pattern, by shift-clicking on some point. These will be marked by small green circles, and mean that in the search these points may be either empty or contain a stone of either color.

For example, the following pattern finds all kos (that are not on the edge):



2.4 Game Info search

If you are looking for games by a particular player, from a particular event or from a certain time period, you can use the game info search.



The games have to match all the requirements (Black Player, Event, ...) simultaneously. The corresponding string has to occur at the beginning of the data, but you can use the percent sign `%` as a wildcard, i.e. if you enter '% Chikun' as player, games where Cho Chikun played will be found.

The 'Anywhere' entry is simply a text search in the SGF file. This allows you to search for the result (use 'RE[W' or 'RE[B'), for games which have a game comment (use 'GC['), etc.

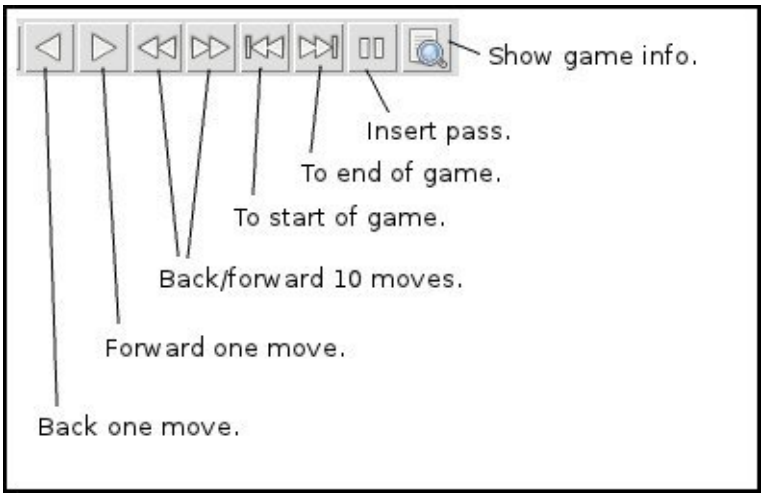
2.5 The SGF editor

You place stones by clicking (with the left mouse key) on an intersection. The four left-most buttons above the board control if you play black/white (resp. white/black) stones alternatingly, or if you place black (resp. white) stones, in order to set up a position.

In order to delete stones or to place labels, you have to select the appropriate tool among the 'edit tools' in the data window. Then you can perform the corresponding operation by holding down the Control key and clicking on an intersection.

With Shift + right-click you can go to the node where some move was/will be played.

Kombilo's main board has two more features which are related to the pattern search: You can place wildcards on the board (resp. delete them) by shift+click, and you can select the relevant area for the pattern search by clicking the right mouse key, and dragging.



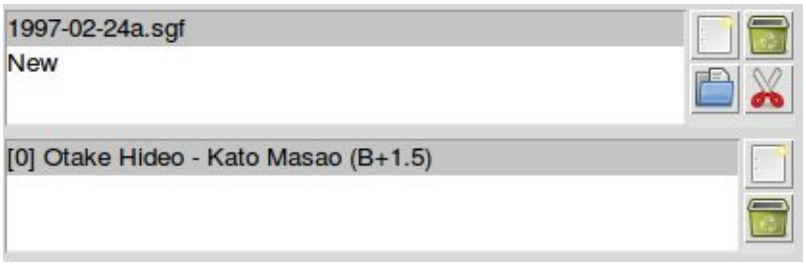
The navigation keys above the board let you move around in the current game record: one move back, one move forward, 10 moves back, 10 moves forward, to the beginning resp. to the end of the current game. All these can also be done by using keys: left, right, up, down, home, end.

If the current SGF file contains variations, you can switch between the alternatives for the current move with the PageUp and PageDown keys.

You can also use the SGF editor without the database functionality by starting the program `v.py`.

2.5.1 The SGF data column

File list



At the top of the right hand column, there is a list of all SGF files that have been loaded during the current session. The currently active file is highlighted; you can change that by clicking on another item in the list. The buttons on the right let you create a new file, open a file from disk, delete a file, or split a collection. Deleting a file just means deleting it from this file list. The file on your disk will not be deleted. Splitting a collection serves to split an SGF file which contains several games into many files with one game each. You will be asked for a filename, and the files will then be saved under the names filename0.sgf, filename1.sgf, filename2.sgf, etc.

If changes have been made to a file after it has been saved, the file name is preceded by a *.

> **Warning:** By default, Kombilo will not ask you if you want to save the changes, so you have to pay attention to the *, and save the files yourself, if you want to keep the changes! You can change this behavior by selecting the *corresponding option*.

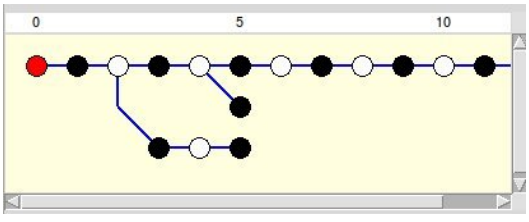
Game list

Below the file list there is a list of game records in the current SGF file. (Usually SGF files contain just a single game, so chances are that you will never use this pane. You can just minimize it to height 0.) You can select games by clicking on them, and change the order by drag and drop. The buttons on the right let you create new games and delete games from the list.

Game info

This shows part of the game information (names of players, result, date, etc.) of the current game. In order to see the full game information, or to edit it, use the button depicting a looking glass and a sheet of paper above the go board.

Game tree



Here the tree structure of the current game is shown. Nodes with a black/white move are shown black resp. white; others are red. Nodes with a comment or a label on the board have a small blue dot in the center.

The green mark shows the current move (i.e. it corresponds to the position currently shown on the main board).

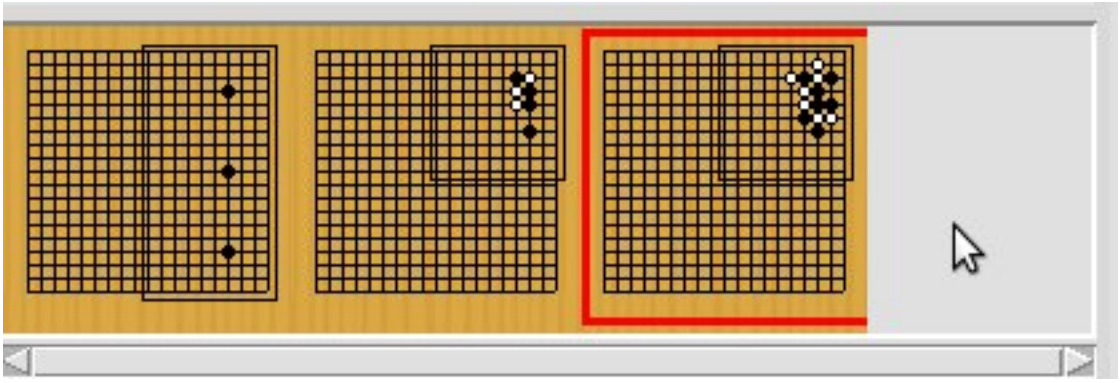
By clicking on a node, you can go to the corresponding move.

Comments

In this window the comments which the SGF file contains for the current node are displayed.

Kombilo: Search history

This frame contains a list of previous search patterns. Click on one of the small boards to go back to the corresponding pattern search (i.e. the pattern and the game list are restored to what they have been right after the search).



A right-click on one of the board brings up a small menu, which lets you delete that entry, put the entry on hold resp. release it.

2.6 The game list column

At the top, the game list window shows the number of games currently in the list, and the B/W winning percentages (the two numbers will often not add up to 100% since there might be Jigo's, unfinished games etc.)

Right below the list, there is a frame where (part of) the game information for the currently selected game in the list is shown (just click on a game to select it).

At the bottom, there is a "notebook" with one sheet ("tab") each for the pattern search statistics, the pattern search options, the game info search, the date profile, tags and for messages.

Right above the notebook, there is a toolbar with several buttons and switches.

The 'home' button resets the game list, so that it includes all the games in the database again. The 'search' button starts a pattern search. The 'back' button jumps back to the previous search: the position on the board is restored as well as the game list. (Previous search patterns are also shown on small boards in the "History" frame of the data window.)

With the button depicting a labeled white stone, you can display the labels showing the continuations in the current search pattern (resp. remove them again).

The button depicting a mouse toggles the *1-click mode*. If this mode is active, every click on the board triggers a search. That can be quite practical in order to play through joseki sequences, say. If this mode is inactive, single

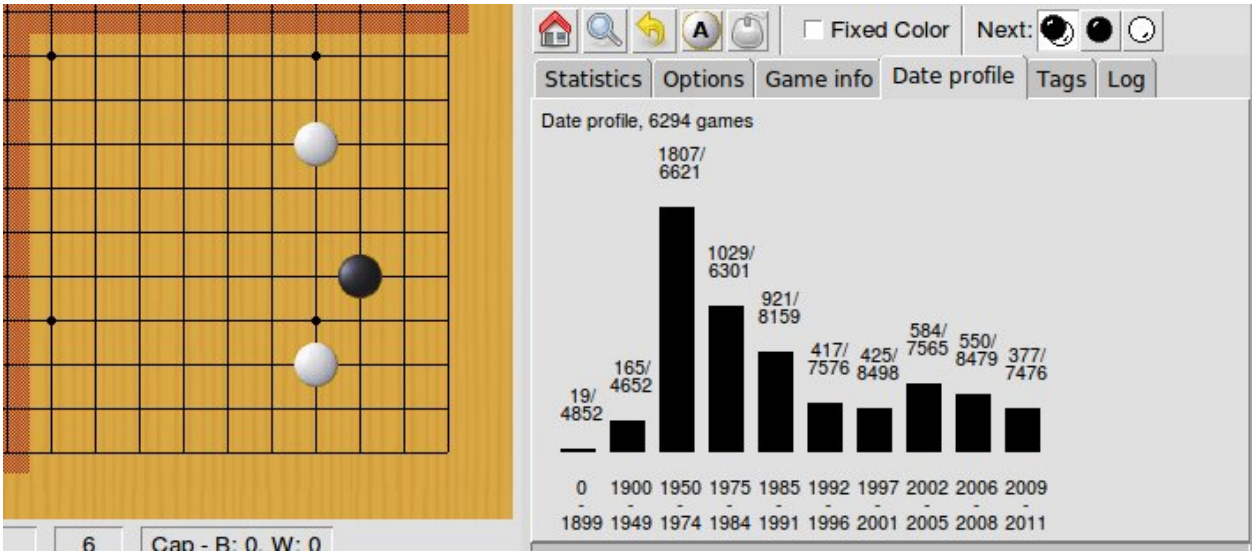
clicks will just place a stone on the board. In this case, you can place a stone and start a search at the same time by double-clicking.

Check the *fixed color* checkbox to disable searching for patterns where black/white are exchanged. Use the *Next* option to specify that either player or black or white should move next in the selected area.

In the game info search window, you see entry fields for the search criteria: white/black player, player, event, etc. If you select the 'Referenced' option, only games with a reference to a commentary will be shown. The "clear" button clears all entries; the back and forward buttons restore the entries from previous searches. Unlike the back button for pattern searches, they do not change the game list. Last but not least, there is the button to start a search; you can also start the search by pressing Enter in one of the entry fields.

The **Go to: field** makes it easy to find specific games in the game list quickly. The 'Go to' entry always works with respect to the current sort criterion. Let's assume that you sorted the database by date. Then entering something in the 'Go to' field will jump to the closest game in the game list the date of which starts with what you entered.

2.6.1 Date profile of the database



The *Date profile* tab shows you how the games which are currently in the game list are distributed over time. The height of each bar shows the proportion of games in the current game list with respect to all games in the database in the same time period. Say you do a pattern search, and then select the *date profile* tab. If one bar is twice as high as another one, then this means that in the first time period the pattern was played twice as much as in the second one. The height of the bars does not contain information about the absolute number of games in the current game list. However, these numbers are printed above the bars (number of games in current list/number of games in whole database).

Computing the date profile is pretty slow (much slower than a pattern search), so you should keep this tab open only as long as you are really interested in the results.

2.6.2 Tags

You can tag games in order to find them more easily and to carry through more complicated searches.

The *Tags* tab lists all existing tags. The following ones are built into Kombilo and are set (semi-)automatically:

- Handicap game; set automatically for all handicap games.
- Professional (a game where at least one professional player plays). You can choose during processing whether and in which way Kombilo should set this tag.
- Reference to commentary available; set automatically for all games for which a reference to a game comment in the literature is available. You can configure which books/journals should be considered here by editing the file `kombilo.cfg` accordingly.
- Seen: set automatically for all games which you opened in the SGF viewer.

If you select a game in the game list, the tags which it carries are highlighted in the tag list. On the other hand, you can specify how tagged games should be marked in the game list (text color/background color).

Tag search

The tags in the tag list have an *abbreviation* which is written in square brackets on the left hand side of the entry. You can search for tags using these abbreviations, and combining them using the logical operators `and`, `or`, `not`, and parentheses. So for example:

- **H** searches for all handicap games.
- **S and C** searches for all games you have previously opened, and for which a reference to a commentary is available.
- **A and B and not C** searches for all games which carry the tags A and B, but not the tag C (assuming that you created these tags before; see below).

Just enter the search expression into the entry field below the tag list and press enter, or click the looking glass button right of this field.

Creating new tags/deleting tags

To create a new tag, add its abbreviation (which must not yet be taken) followed by a space and the description of the tag, like this:

```
N My new tag
```

and click the button showing a plus sign.

To delete a tag from the tag list (and hence to remove it from all games), enter its abbreviation and click the button showing a minus sign.

Setting/removing tags on games

To specify the tags of a **single game**, select the game in the game list. The tags which it currently carries are highlighted. You can now select/deselect tags in the tag list by clicking them (use Control-click to select multiple entries). To set the chosen combination of tags on the selected games, click the second button from the left in the tags toolbar.

To add a tag to **all games currently in game list**, enter its abbreviation into the text entry field, and click the third button from the left. To remove a tag from all games currently in the game list, enter its abbreviation into the text entry field and click the fourth button from the left (depicting a broom).

For instance, you could create a tag `A Large Avalanche Joseki`, do a pattern search for the large avalanche joseki, and tag all games in the resulting game list with the tag `A`. The you can easily search for all these games, also in combination with other tags, and you can search for all games where the large avalanche does not occur, by searching for `not A` - and again, this can be combined with searching for other tags.

2.7 Analyzing a game

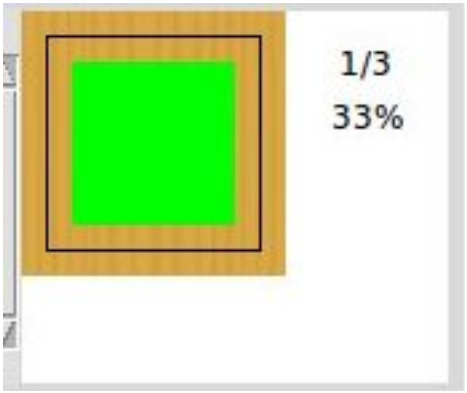
If you want to analyze a game of your own, just load it into the main board with the 'Open' command in the file menu (or use the 'Open' button next to the file list in the data window). Use the navigation buttons to navigate through the file, and search for patterns appearing in your game: for the first few moves you may want to do a whole board search, in order to see up to which point the fuseki you played also occurs in professional games, and afterwards you have to select an appropriate relevant region.

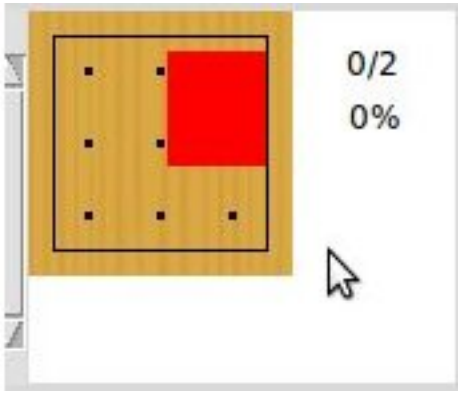
You can also load a fuseki or joseki dictionary For example, Kombilo works quite well with Kogo's joseki dictionary. To navigate all the variations, you should enable the 'Show next move' option.

2.8 Guess next move mode

One fun way to study go is to replay professional games by guessing the next move. If you click on the corresponding button in the SGF edit toolbar in the data window, you enter Kombilo's guess mode. That means that clicks on the board will be interpreted as guesses - if it coincides with the next move in the current SGF file, that move is played; otherwise no stone is placed on the board.

When you switch to the 'guess next move' mode, a small frame appears next to the game tree, which gives you some feedback on your guesses. If your guess is right, it displays a green square (and the move is played on the board).





If the guess is wrong, it displays a red rectangle; the rectangle is roughly centered at the position of the next move, and the closer your guess was, the smaller is that rectangle. Furthermore the number of correct guesses and the number of all guesses, as well as the success percentage are given.

Of course, if you just can't find the next move, you can always use the 'Next move' button above the board.

2.9 Further notes

2.9.1 Using Kombilo with non-latin (Unicode) characters

Kombilo works out of the box with UTF-8 encoded SGF files, but currently not with other encodings.

2.9.2 How can I reset the correct/wrong counter in the "guess next move" mode?

Currently, you can only reset the counter by quitting and reentering the "guess next move" mode.

CHAPTER

THREE

KOMBILO MANUAL

3.1 Installation

3.1.1 Linux

The following instructions cover the installation of Kombilo under Ubuntu Linux (current version, i.e. 11.10). If you use another flavor of Linux and are somewhat familiar with it, you will easily adapt them.

Quick start: installation on a Ubuntu system

With the following commands you can install Kombilo on a Ubuntu system. Lines starting with a # are comments - no need to type them. These instructions will create a subdirectory `kombilo` inside the current directory.

There are three main steps to the installation: installing Python and the Python packages, downloading the Kombilo files and extracting them, and compiling the extension for the fast pattern search. See below for more details on the different steps.

```
# Install the packages that Kombilo depends on (and wget for the next step):
sudo apt-get install python-tk python-imaging python-imaging-tk python-pmw
sudo apt-get install python-configobj g++  libsqlite3-dev
sudo apt-get install python-dev libboost-filesystem-dev libboost-system-dev

# download the Kombilo archive
wget https://bitbucket.org/ugoertz/kombilo/downloads/kombilo-0.7.4.tar.gz

# unpack the archive
tar xfz kombilo-0.7.4.tar.gz kombilo

# compile the C++ extension
cd kombilo/lk
python setup.py build_ext
cp libkombilo.py build/lib.linux-*/_libkombilo.so ../src/

# start the program
cd ../src/
./kombilo.py
```

Now continue with the *Getting started* section of the tutorial. After installing, you start the program by executing the `kombilo.py` script in the `kombilo/src` directory.

Basic dependencies

The best Python version to run Kombilo on is **Python 2.7**. You might be able to get it to work with 2.6, but this will need some more work (at least you need to install pyttk separately). It is currently not compatible with Python 3.

Unless you are a Python specialist, the easiest way to install the packages required for Kombilo is to install the following packages using the package manager of your choice (`synaptic`, `aptitude`, `apt-get` etc.):

```
python
python-tk
python-imaging
python-imaging-tk
python-pmw
python-configobj
```

If you *are* a Python specialist and want to retain finer control (and place Kombilo in a virtualenv environment, say), it is enough to install the `python` and `python-tk` packages, and then to use `pip` to install the Python packages specified in the `requirements.txt` file. In addition, in this case, you have to install the Python Mega-Widgets by hand: download the tar.gz file from http://pmw.sourceforge.net/, unpack and install using python setup.py install.

Downloading Kombilo

tar.gz files

Download the `kombilo-0.7.4.tar.gz` archive from the Kombilo downloads site.

Unpack the archive somewhere by

```
tar xfz kombilo-0.7.4.tar.gz kombilo
```

This will extract all the files into the kombilo subdirectory.

Mercurial repository

You can also clone the Kombilo mercurial repository. See *Development* below for some details.

Libkombilo

To compile the extension for the pattern search, make sure that the following packages are installed:

```
g++
python-dev
libboost-filesystem-dev
libboost-system-dev
libsqlite3-dev
```

Then, to compile the package, do the following:

```
cd ~/go/lk
python setup.py build_ext
cp build/lib.*/_libkombilo.so ~/go/kombilo/
```

Development

If you want to work on Kombilo or Libkombilo yourself, you can clone the mercurial repository:

```
hg clone https://bitbucket.org/ugoertz/kombilo
```

Make sure (before ...) that you have mercurial installed, and also install SWIG:

```
sudo apt-get mercurial swig
```

Before you can compile the libkombilo extension, you need to run swig:

```
cd kombilo/lk
swig -c++ -python libkombilo.i
python setup.py build_ext
cp libkombilo.py build/lib.linux-*/_libkombilo.so ../src/
```

Build the documentation

If you installed Kombilo from a `tar.gz` archive, then you can skip this step. If you installed directly from its Mercurial repository, and want to use the documentation offline (either directly or from the Kombilo Help menu), then you need to build the documentation yourself. If you install it from a tar.gz file, then you can skip this step.

Kombilo documentation

Install Sphinx either via `pip install sphinx`, or globally by

```
sudo apt-get install python-sphinx
```

and in the `doc/` directory, run

```
make html
```

to build the HTML documentation (to be found in `doc/_build/html/`), or

```
make latexpdf
```

to build a pdf file. (For the latter, you need to have LaTeX installed on your computer).

Libkombilo documentation

Install Doxygen by

```
sudo apt-get install doxygen
```

and in the `lk/doc/` directory, run

```
doxygen
```

Besides a lot of warnings, this will generate HTML and LaTeX files of the documentation in `lk/doc/build/`.

3.1.2 Windows

Installer

The installer installs the Kombilo package together with all libraries etc. which it depends on. Using it should allow you to ignore the whole Installation section of this documentation.

If you would like to know the details, here is some further information:

Basically, the installer extracts an archive which contains the Python interpreter, further packages that Kombilo depends on, and the Kombilo files themselves to your hard disk. In this way, for one thing you do not have to install all these packages yourself, and furthermore Kombilo will not interfere with different versions of these packages that you might have in use.

Main kombilo directory: The Kombilo files all go into the installation directory that you can specify during installation; typically `c:\Program Files\kombilo07` or something similar

Source code: The Kombilo source code is included as a zip archive in the main Kombilo directory.

Microsoft DLLs: Python, and hence the Kombilo installer, relies on a couple of DLLs (shared libraries) that are part of Microsoft's Visual C++ compiler package. The installer includes a self-extracting archive which may be freely distributed; if you do not yet have them, the DLLs will be installed on your system, in an appropriate folder.

Configuration/log files: The individual configuration file `kombilo.cfg`, and (if necessary) the error log file `kombilo.err` will be written to a directory inside the *APPDATA* directory (something like `c:\Users\yourusername\AppData\Roaming\kombilo\07\`).

Uninstall: The installer creates an *uninstall* menu entry in the Kombilo menu inside your start menu (unless you disable the start menu entry altogether). The uninstaller will remove all files that Kombilo created inside the main kombilo directory, as well as the start menu entry and possibly the desktop icon. It cannot (and should not) remove the DLLs. Neither will it remove the configuration files (see above). This allows you to uninstall kombilo, install a new version, and continue to use your old configuration. Instead of using the menu entry, you can also directly invoke the exe file (its file name starts with `unins`) directly.

Installation from scratch

If you want to build Kombilo from source yourself, here are some notes. The *libkombilo* extension has to be compiled with a C++ compiler. You could (probably, and probably easier) use Microsoft Visual C++, but I used the open source MinGW compiler. To use MinGW, some preparations have to be made:

In `\Python27\Lib\distutils\`, create a file `distutils.cfg` with the following content:

```
[build]
compiler = mingw32
```

Furthermore, there is a problem with the Python distutils core: it passes the `-mno-cygwin` option to MinGW, but this option is not recognized. One way around this is to remove the `-mno-cygwin` from lines 322, 323, 324, 325 and 326 of `\Python27\Lib\distutils\cygwinccompiler`.

Install sqlite3 (and create a libsqlite3.a file for MinGW) and the Boost library (only the header files are needed for libkombilo; there is no need to compile the boost library).

After that, you should be able to run `python setup.py build_ext` in the `lk` subdirectory inside your Kombilo directory.

After installing Python and the packages (configobj, PIL, Pmw) that Kombilo depends on, you should now be able to run `python kombilo.py`.

To create a stand-alone exe file, you can use py2exe. To distribute the whole thing as a one-file-installer, I use InnoSetup. See also the `deploy_win` method in the fabric file `fabfile.py` in the main Kombilo directory.

3.1.3 Mac OS X

Kombilo runs on Macs, and since Mac OS X is a Unix variant, most of the notes in the *Linux* section apply to Mac OS X, as well. However, under some circumstances there appear to be some problems, depending on the versions of the packages that Kombilo depends on. Simon Cozens reported that on a Mac (with Mac OS X 10.6) with Homebrew he could run Kombilo after

```
sudo easy_install configobj setuptools pyttk pip
brew install PIL boost
sudo pip install pil
```

then installing Pmw from source and building the libkombilo extension via `python setup.py build_ext` as described in the *Linux* section.

On the other hand, sometimes the Python Imaging Library PIL seems to cause problems (installing it via Homebrew seems to be the best way). In fact, it is used only for the nicer stone pictures, so it is not too bad to not use it, and I

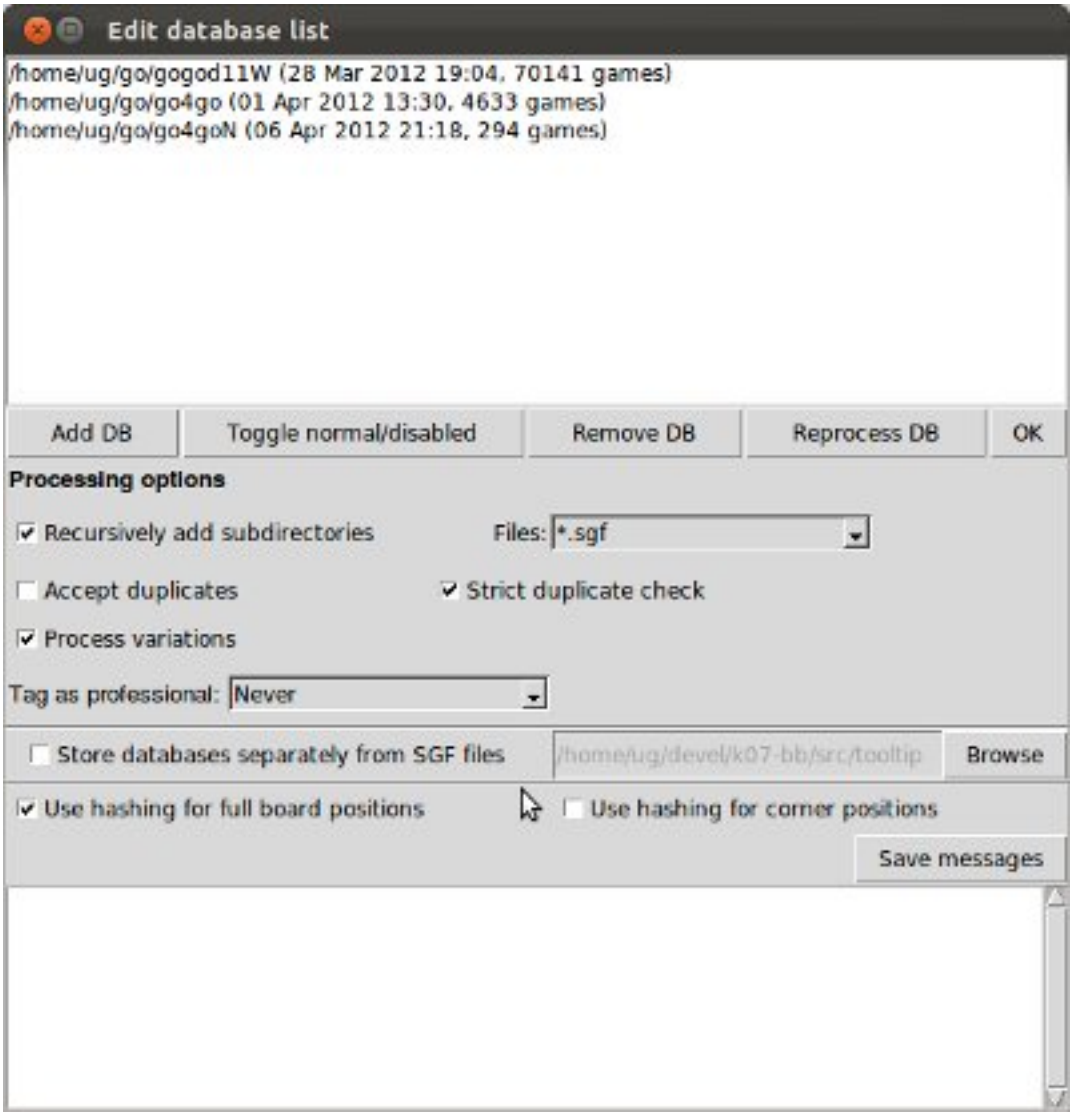
made this the default for Macs. Change the *corresponding option* if you do want to use it. (Thanks to R. Berenguel for his help with figuring this out.)

If you have Python 2.6, you need to install the `pyttk` package to run Kombilo. In Python 2.7, which is the preferred Python version for Kombilo, this package is already included in Python.

See also the *Only one mouse button* option.

3.1.4 Setting up the SGF databases

Before you can start working with Kombilo, you need to add your SGF files. For Kombilo, a database is just a directory with SGF files in it. Select `Edit DB list` in the `Database` menu. A new window will open.



Add databases

In the lower section *Processing options* you can select which kind of files you want to add, whether to recursively add all subdirectories, whether to accept duplicates, and whether to store variations in the database for pattern search. You

can also select whether all games (or none) of the database should be considered as pro games, or whether this should be decided by the rank specified in the files.

If you prefer, you can specifiy a folder where the Kombilo files should be stored. If you do not name a folder here, the files will be stored in the folder containing your SGF files.

Finally, you can choose which algorithms you want to use with your databases. (You can also *disable the hashing algorithms* for each pattern search, but you can only use then if you selected the corresponding option before processing the games.)

The hashing algorithms speed up searches for full board and corner positions respectively, on the other hand the procesing takes slightly longer, more disk space is consumed, and Kombilo uses more memory when running.

Messages during processing

In the lower text area, Kombilo will output messages about the processed games.

- **Duplicates**: Games which are duplicates to games already in the database are named. Being a duplicate is tested with the method chosen in the options. In every case, the Dyer signature (position of moves 20, 31, 40, 51, 60, 71) is compared. With strict duplicate checking, in addition the final position is compared. See *Find duplicates*.
- **SGF Error**: If there was an SGF error, Kombilo issues a warning. It tries to do its best to recover, and will insert as much of the game as it understands into the database anyway.
- **Unacceptable board size**: Currently, Kombilo processes only 19x19 games.
- **not inserted**: For games which are not inserted into the database, this message is appended to the error message. Otherwise, the game is inserted.

File sizes

No Hashing: roughly 170 MB for about 70,000 games (GoGoD winter 2011)
Hashing for full board positions: roughly 270 MB
Hashing for full board and corner positions: roughly 365 MB

After adjusting the options, if necessary, select `Add DB` in order to add some SGF files.

The optimal size (i.e. number of SGF files) of the databases depends mostly on the amount of memory in your computer. I recommend a size of at least 1,000 - 2,000 SGF files per database; that should be fine on almost every system. If you have a lot of memory, you can experiment with larger databases to increase performance. For databases with ten thousands of games, the "finalizing" will take quite some time (a few minutes for the 70,000 GoGoD games on my laptop), so please be patient.

Kombilo will create several database files: `kombilo.db`, `kombilo.da`, and if you use the hashing algorithms, also `kombilo.db1` and `kombilo.db2`.

Toggle normal/disabled

If you want to temporarily exclude a database from some searches, select it and use this button to set its status to 'disabled'. It will then be marked as 'DISABLED' in the database list. Its games will not show up anymore in the game list, and will not be found by any search. Nevertheless, Kombilo's database files written during the processing are still available, and if you toggle the status back to 'normal', you can use that database again without processing it again.

Remove a database

If you want to remove a database from Kombilo's list completely, select it and press this button. The database files Kombilo has written will then be deleted. Of yourse, the SGF files themselves will not be deleted (Kombilo will actually never change them.) If you want to add this database again later, it will have to be processed again.

Reprocess a database

If you made any changes to the SGF files in one of the database directories (or added/deleted SGF files in there), you should reprocess the database, so that the pattern search really uses the information corresponding to the current version of the SGF files.

Since version 0.7.1, reprocessing keeps all the tags on your database. This is usually the desired behavior. If you prefer to have all tags deleted, instead of reprocessing, remove the databases and then add them again.

Save messages

If there are errors in the SGF files, or if Kombilo finds duplicates, a message is issued. The 'save messages' button allows you to save these messages into a file, such that you can look at them later again in order to correct the errors. (After correcting any errors, you should reprocess the corresponding databases.)

Further notes

With Ctrl-click and Shift-click you *can select several databases* in the list simultaneously. The "Toggle normal/disabled", "Remove" and "Reprocess" buttons will then apply to all the selected databases.

Currently it is not possible to add single games to a database, or to delete single games.

3.2 Searching

There are two main ways to search in your database: by patterns occurring in the games (*Pattern search*), and by properties written out in the SGF file (such as the players, the result, the date, the event where the game was played etc.). We call the latter type of search a *Game info search*.

Furthermore, you can search for tags - either games that were automatically tagged by Kombilo (e.g. handicap games), or for games that you tagged yourself - (*Tag search*), and for the Dyer signature of a game (*Signature search*). This is typically used less often, but may be useful to quickly find a game whose Kifu you have in printed form.

3.2.1 Pattern search

Enter the pattern you want to search for by "putting down" the black and white stones on the board, and select the size of the pattern (the "relevant region" for the search) by clicking with the right mouse button and dragging.

Search options

fixed color If this is set, the pattern is searched only as it is given on the board. Otherwise, the pattern with black and white exchanged is also considered. In the list of results given at the end of each line in the game list, hits where the colors are exchanged are marked by a minus sign following the move number.

next move Specify whether black or white should move next in the search region.

fixed anchor Do not "move" the pattern along the side or within the center of the board.

Search in variations Usually, Kombilo searches for the pattern in all variations in the game. If you switch this off, only the first ("main") variation will be considered.

move limit Find only occurrences before the given move number. The maximum value 250 means: find all occurrences.

algorithms Choose whether Kombilo should use hashing algorithms for full board patterns and/or for corner patterns. (If you want to use them, you have to choose them when creating the database from your SGF files.)

Wildcards

You can put down a wildcard by shift-clicking. A green dot means that this spot may either be empty, or contain a black stone, or contain a white stone. A black dot means that the spot may be empty or contain a black stone, and analogously for a white dot. You can go from empty to green, black, white, etc. by shift-clicking several times.

Move sequences

You can search for move sequences, i.e., specify that some stones of the pattern have to be played in a certain order. To do so, first create the final pattern of the sequence. Then put numbers as labels on those stones that constitute the sequence that must have been played to arrive at this pattern. You can leave stones unnumbered - this means that they have to be present in the results before the move sequence starts.

> **Warning:** Currently there is no good way of dealing with captures, i.e., if a stone of your sequence captures other stones, you cannot search for the sequence with the current mechanism. This is only a problem of the user interface; a mechanism of telling Kombilo about the captured stones is currently missing (and will hopefully be added some time).

Further notes

> **Warning:** Passes in the game
> In the unlikely case that one of the players passed in the middle of the game (but see file 1998-04-21a in the GoGoD database), the handling of continuations is not consistent between the different algorithms.

3.2.2 Game info search

In the game info search tab, you can search for properties of the game which are written out in the SGF file.

For all text search fields (except for *Event*, *Anywhere*, *SQL*), Kombilo returns all games where the corresponding game info starts with the given string; i.e., if you search for *Cho* as player, you will get games by *Cho Chikun* as well as *Cho U* (and all other Cho's).

For the *Event* and *Anywhere* fields, all games are returned where the given text occurs anywhere in the event field or in the whole SGF file, respectively.

You can in addition use the percent sign `%` as a wildcard yourself, e.g.: if you search for *%Hideki* as the player, you will get all games of *Matsuoka Hideki* as well as those of *Komatsu Hideki* etc.

Player matches black player and white player names.

from, to Specify dates in the form `YYYY` or `YYYY-MM` or `YYYY-MM-DD`. If you want to search for a date in a different form, you need to use the *Anywhere* or the *SQL* search field.

SQL This is passed directly to the database as the `WHERE` clause of an SQL statement. Examples:

```
not PW like 'Cho%'
DATE < 1900-00-00 or DATE >= 2000-00-00
```

The column names of the SQL table are

```
PB (player black)
PW (player white)
RE (result)
EV (event)
DT (the date as given in the sgf file)
date (the date in the form YYYY-MM,DD)
filename
sgf (the full SFG source).
```

In SQL statements, you have to take care of *escaping* characters yourself; inparticular, single quotes occurring inside the search string must be doubled:

```
PB = 'Yi Ch''ang-ho'
```

3.2.3 Tag search

The tags in the tag list have an *abbreviation* which is written in square brackets on the left hand side of the entry. You can search for tags using these abbreviations, and combining them using the logical operators `and`, `or`, `not`, and parentheses. So for example:

- **H** searches for all handicap games.
- **S and C** searches for all games you have previously opened, and for which a reference to a commentary is available.
- **A and B and not C** searches for all games which carry the tags A and B, but not the tag C (assuming that you created these tags before; see below).

Just enter the search expression into the entry field below the tag list and press enter, or click the looking glass button right of this field.

3.2.4 Signature search

In order to check for duplicates in the database, Kombilo computes a modified Dyer signature of every game in the database. The signature of a game is given by the coordinates (in SGF format) of the moves 20, 40, 60, 31, 51, 71. This almost always characterizes a game uniquely.

In order to detect games which differ only by a symmetry of the board, Kombilo uses a symmetrized Dyer signature: the Dyer signatures for the game and for all rotations/reflections of the game are computed, and then the smallest of these (with respect to the lexicographic order) is stored.

You can also search for the signature. This might be useful to see if a certain game is in the database if you have the game record in some (foreign-language) book, say, and cannot read the player's names.

Select *signature search* from the database menu, and a window will pop up, where you can enter the coordinates of the corresponding moves. If you click on an intersection on the board, the corresponding coordinates will be entered in the currently active text entry below, and the next entry will be made active. So you can enter the signature simply by clicking on the places where moves 20, 40, ... were played. You can also omit some of them (in most cases, two or three of the moves will be enough to characterize a game uniquely).

You can print the signature of a game to the log tab by selecting it in the game list and pressing *s*.

3.2.5 Export search results

If you want to save some information on a pattern search, you can use the 'Export search results' function in the Database menu. This will open a new window with a very simple text editor. It will contain the search pattern, the search pattern with the continuations, some statistical information on the search, and the number of hits in each database.

You can edit the information and in the end save the text to a file. I would be interested in hearing your opinion if other or additional information should be given, or if the information should be presented in another format.

Before the text editor opens, you will be asked if you want "ASCII" or "Wiki" style output. Usually you will choose 'ASCII', which produces plain text. If you want to use the output for Sensei's Library, choose 'Wiki' instead. You can also choose if all continuations, or if only ten of them should be displayed.

The text editor has a button which lets you include the complete current game list (names of players, etc.).

3.3 The game list

The game list shows the current list of games. Depending on your configuration, it shows the *white player*, the *black player*, the *result*, the *date*. In the options menu, you can choose to include (or exclude) the *file name* as the first item, and the *date* as the last item.

After a pattern search, the game list shows a list of hits for each game: the move number when the pattern occurred; the continuation (if any); a minus sign if the pattern occurred with black/white exchanged.

Entries with different color (or background color) reflect tags set on games. This behavior can be configured in kombilo.cfg.

3.3.1 Statistics

The statistics tab shows information about the continuations in the most recent pattern search. For each of the 12 most common continuation, a bar indicates the frequency. The black/white parts of the bar indicate the number of times that black/white played in the pattern region immediately after the pattern was completed. The dark gray/light gray parts indicate the number of times that black/white played in the pattern region after a tenuki.

3.3.2 Date profile

The bar diagram shows the distribution of games in the current list in comparison to all games in the database, by date. The height of the bars indicate the proportion of games in current list versus games in complete database. *The height of the bars does not contain absolute information*, i.e. even if there are only very few games in the current list, the highest bar will have full height. Absolute information is printed above the bars (number of games in current list in this time period/number of games in complete database in this time period).

Computing the date profile is pretty slow (much slower than a pattern search), so you should keep this tab open only as long as you are really interested in the results.

3.3.3 Tags

You can tag games in order to find them more easily and to carry through more complicated searches. The *Tags* tab lists all existing tags. The following ones are built into Kombilo and are set (semi-)automatically:

- Handicap game; set automatically for all handicap games.

- Professional (a game where at least one professional player plays). You can choose during processing whether and in which way Kombilo should set this tag.
- Reference to commentary available; set automatically for all games for which a reference to a game comment in the literature is available. You can configure which books/journals should be considered here by editing the file `kombilo.cfg` accordingly.
- Seen: set automatically for all games which you opened in the SGF viewer.

If you select a game in the game list, the tags which it carries are highlighted in the tag list. On the other hand, you can specify how tagged games should be marked in the game list (text color/background color).

Creating new tags/deleting tags

To create a new tag, add its abbreviation (which must not yet be taken) followed by a space and the description of the tag, like this:

```
N My new tag
```

and click the button showing a plus sign.

To delete a tag from the tag list (and hence to remove it from all games), enter its abbreviation and click the button showing a minus sign.

Setting/removing tags on games

To specify the tags of a **single game**, select the game in the game list. The tags which it currently carries are highlighted. You can now select/deselect tags in the tag list by clicking them (use Control-click to select multiple entries). To set the chosen combination of tags on the selected games, click the second button from the left in the tags toolbar.

To add a tag to **all games currently in game list**, enter its abbreviation into the text entry field, and click the third button from the left. To remove a tag from all games currently in the game list, enter its abbreviation into the text entry field and click the fourth button from the left (depicting a broom).

For instance, you could create a tag `A Large Avalanche Joseki`, do a pattern search for the large avalanche joseki, and tag all games in the resulting game list with the tag `A`. The you can easily search for all these games, also in combination with other tags, and you can search for all games where the large avalanche does not occur, by searching for `not A` - and again, this can be combined with searching for other tags.

Importing/exporting tabs

You can export the tags in your current database, and import them later to a (different) database. (Use the corresponding menu items in the Database menu.) The games are identified by the Dyer signature and some additional hash code, so the imported tags will be set precisely on the games *with the same moves* as the games that carried the tags when exporting.

In version 0.7, you can/should use this to transfer your tags when updating your database by reprocess. Since version 0.7.1, reprocess does this for you automatically.

3.3.4 GoTo field

Use this field (in the game info search tab) to jump to a game in the game list quickly by entering a few letters of the current sort criterion (see the options/game list menu). E.g., if you sort the games by date, entering `1990` will bring

you to the games from 1999; if you sort the games by white player, entering `Cho` will bring you to the games with white player Cho.

3.3.5 Log

In this tab, Kombilo prints out some information about its actions (timing of searches etc.).

3.3.6 Find duplicates

Use `Find duplicates` in the `Database` menu to produce a list of duplicates in the database (or rather, in all the databases that are currently active). The list will be presented in a new window and can be saved as a text file. The duplicate check will be strict (i.e., the Dyer signature and the final position will be compared) or non-strict (only the Dyer signatures will be compared) depending on the setting of the corresponding processing option. This option can be changed in the `Edit DB list` window or in the `Options-Advanced` menu.

3.4 The SGF editor

Most of the SGF editor handling should be self-explanatory, so this section is rather brief.

Warning: By default, Kombilo does not ask for a confirmation before discarding unsaved changes, or before deleting a game. You can change this in the options menu, or in the `kombilo.cfg` configuration file.

3.4.1 Guess mode

Activating the *guess next move* button (depicting a question mark) in the SGF edit toolbar in the data window starts Kombilo's guess mode. That means that clicks on the board will be interpreted as guesses - if it coincides with the next move in the current SGF file, that move is played; otherwise no stone is placed on the board. For obvious reasons, the *show next move* option will be disabled as long as the guess mode is active..

When you switch to the 'guess next move' mode, a small frame appears next to the game tree, which gives you some feedback on your guesses. If your guess is right, it displays a green square (and the move is played on the board).

If the guess is wrong, it displays a red rectangle; the rectangle is roughly centered at the position of the next move, and the closer your guess was, the smaller is that rectangle. Furthermore the number of correct guesses and the number of all guesses, as well as the success percentage are given.

If you just can't find the next move, you can always use the 'Next move' button above the board to move forward in the game.

3.4.2 Export current position/SGF

Similarly to the *Export search results* function, you can "Export current position" (in the database menu): this will open a text editor with the current position. Again, you can choose "ASCII" or "Wiki" type. In addition, Kombilo can put the next moves (up to 9 moves) on the board, marked by the numbers 1 to 9.

Finally, you can also export the SGF source of the current game (see the File menu), in a text editor.

3.4.3 Miscellaneous remarks

With the **rotate/flip SGF file** menu items (in the Edit menu), you can rotate and flip the game; th SGF file is changed so as to describe the game with the new orientation. This is useful if you want to change a game record to obey the usual convention that the first move is in the upper right corner.

With the **split collection** button (depicting scissors) right to the list of files, you can split one SGF file containing several games into a collection of files, one for each game.

With *Copy current SGF files to folder* in the Database menu you can copy the SGF files corresponding to the games currently in the game list to some folder (e.g. in order to use them with a different program).

@@monospace in SGF comments. If you put the string `@@monospace` as the first line of a comment of an SGF node, Kombilo will display the comment in a fixed width font. This is useful whenever you want to output tabular data in a node (see the `sgftree` script).

In the **Game info** edit window, in the *Other SGF tags* entry field you must enter correct SGF code, i.e. special signs such as `]` and `\` must be escaped by a preceding `\`.

3.5 Key and mouse bindings

3.5.1 Global key bindings

- `Control-r` reset game list
- `Control-s` select statistics tab
- `Control-o` select options tab
- `Control-g` select game info search tab
- `Control-d` select date profile tab
- `Control-t` select tags tab
- `Control-p` start pattern search
- `Control-b` go back to previous search
- `Control-e` print information about previous search pattern to log tab

If the *search-history-as-tab* option is 1, then there is also

- `Control-h` select search history tab

3.5.2 Board key bindings

- `Left`/`right`: back/forward 1 move
- `Up`/`down`: back/forward 10 moves
- `Home`/`end`: to start/end of game
- `PgUp`/`PgDown`: navigate variations
- `Control-i`: open game info

3.5.3 Game list key bindings

- `Up`/`down`/`PgUp`/`PgDown`: move in game list
- `Home`/`End`: scroll to left/right
- `Return`: open selected game in viewer
- `Control-a`: print Dyer signature of selected game to log tab

3.5.4 Mouse bindings

- Use Left-click to put stones on the board.
- With Right-click and drag, you select the search-relevant region.
- Use Shift + Left-click you can put (change/remove) *Wildcards* on the board.
- With Shift + Right-clicking on a stone, you can go to the point in the SGF file, where this stone was played.
- The mouse wheel lets you scroll the game list, or scroll through the current game, depending on where the mouse pointer is located.
- The next button triggers a pattern search, the back button goes back to the previous search. (This does not work on Windows.)

3.6 Configuring Kombilo

The most common options can be changed in the *Options* menu. Furthermore, you can configure Kombilo by *editing the file kombilo.cfg* (when Kombilo is not running). Finally, the appearance can be modified by creating/changing the file `kombilo.app` accordingly.

3.6.1 Window layout

You can change the width of the three columns of the main window, as well as the height of the entried in the left hand column by dragging the "sashed" between them to the left/right (or up/down, resp.). Move your mouse pointer slowly over the region between the columns; it should change its look when you are over the sash.

See also the *maximize window* option.

3.6.2 Options in the Options menu

Fuzzy stone placement Place the stones on the main board slightly off the exact point, in a random direction, to make the position look more natural. (Well, some people might think that it is just ugly, so you can switch it off here).

Shaded stone mouse pointer (Don't) Show the current position of the mouse pointer on the board and the color of the next stone to be played by a shaded stone.

Show next move In case a SGF file has been loaded, show the position of the next move with a circle.

Show last move This marks the most recent move with a small circle. Thanks to Bernd Schmidt who provideda a patch for this. (The SGF file is not changed.)

Show Coordinates Show coordinates around the board. **Ask before discarding unsaved changes** If this option is enabled, Kombilo will ask for confirmation before discarding unsaved changes in an SGF file (i.e. before deleting the game from the game list, and before exiting Kombilo).

Jump to match This controls the behaviour of the SGF viewer when you open a game from the game lis tafter a pattern search. If this option is checked, the viewer will jump directly to the position where the pattern you searched for was found in that game.

Smart fixed color If this option is enabled, the 'fixed color' option will be automatically enabled when you select the whole board as search-relevant region, and disabled when you select a smaller region. (You can nevertheless change that after selecting the region and before starting the search.) This is useful because if 'fixed color' is not used, Kombilo regards a position and the same position with swapped colors as equivalent; in the case of whole board searches that can lead to counter-intuitive results when you look at the continuations (e.g. place a black resp. white stone on the upper left resp. upper right hoshi, do a whole board search without 'fixed color', and look at the continuations).

Themes

Kombilo offers you to change its look according to one of a number of themes. Which themes are available depends on your operating system. Just try them out. The effects will be visible immediately.

The 'Game list' submenu

Sorting the game list First of all, in the 'Game list' submenu of the Options menu, you can choose how to sort the game list: by name of white or black player, date or filename.

You can reverse the whole game list by selecting the *Reverse order* option. So if you would like to sort the whole list by date, with the most current games at the top, you could disable 'Sort per database', choose 'Sort by date', and select 'Reverse order'.

Show date/show filename Depending on where your SGF files come from, it might be interesting to include the filename in the game list (as was done automatically in previous Kombilo versions), or to omit it. Similarly, it might be interesting to include the date (if it cannot be read off from the file name, say, or to omit it). These two options allow you to control this. Changing either of these options will reset the game list.

Advanced

Open game in external SGF viewer By default, by double-clicking on a game in a game list, the game is opened in Kombilo's main window. (You can open the game in an external viewer, by shift-clicking, though). If this option is active, double-clicking opens the game in an external viewer (v.py or an alternative SGF viewer). In that case, shift-clicking opens the game in the Kombilo main window.

Alternative SGF viewer If you want to use your customary SGF viewer/editor instead of the viewer coming with Kombilo, enter the command to start it and the command line options that tell it to open a certain sgf file here (put an %f where the filename should be). (If your viewer supports it, you can also put an %n where the move number the viewer should jump to directly should be put.)

If your viewer supports jumping directly to a certain move in a game, you can use %n as a placeholder for the move number of the first hit. Similarly, if your viewer supports SGF collection, you can use %g as a placeholder for the number of the concerning game in the given SGF file.

Under Windows, the file name is put in quotes. This is necessary if the path contains spaces. If you don't want the quotes (or want to set them yourself), you can use %F instead. **Maximize window** (*Windows only*) If this is active, Kombilo will try to maximize its main window on startup. This option will become effective when you start Kombilo the next time (not immediately).

3.6.3 The kombilo.cfg configuration file

All configurable options can be changed by editing the file `kombilo.cfg` in the kombilo folder. This file is a plain text file which you can edit yourself. *You should not edit this file while Kombilo is running.* It is created when Kombilo is started for the first time.

Note: Location of the `kombilo.cfg` file

Depending on your platform, the kombilo.cfg file will be stored in the following place:

Linux/Mac OS: `~/.kombilo/07/`, where `~` is your home directory; on Linux, this is typically `/home/yourusername/`.

Windows: In the folder `kombilo\07\` inside the *APPDATA* folder; typically *APPDATA* is something like `\Users\yourusername\AppData\Roaming\`.

If you want to use several instances of the same Kombilo version at the same time, you can also place the kombilo.cfg file inside the main Kombilo directory. If there is a kombilo.cfg present there, it will be preferred. Note that in this case you need write permissions for this folder.

Lines starting with a # are comments. Most options are explained by comments in this file.

In addition to the options, you can also define how tagged games should be displayed (background/foreground color) in the game list, and which references to commentaries in the literature should be displayed in the game list. **search_history_as_tab** (new in 0.7.1) Set this to 1 in order to put the search history frame as a tab in the right hand column. If the option is 0, then the search history will be displayed as the bottom pane of the left hand column. The current default for this option is 0, in version 0.8 the default will become 1. **use_PIL** (new in 0.7.1) Set this to 0 in order to disable the use of the Python Imaging Library (PIL). If 1, then PIL will be used. If `use_PIL = auto`, then PIL will not be used on Mac OS, but will be used on other systems. This is the default setting, because PIL causes problems on Mac OS X. The only consequence is that without PIL, you will not get the "3D" stones, but just black/white circles as stones. (So if you prefer the flat stones, you could just set this option to 0.)

Uppercase labels If you want to use the 'Export search results' function to produce output for Sensei's Library, it is useful to use lowercase labels for the continuations, since only lowercase letters are automatically understood by Sensei's Library. If you do not want to do that, and find that uppercase labels look better, you can use this option.
Only one mouse button Some Mac OS X users have a mouse with only one button. Using this option, they can mark the search-relevant region with Alt + (left) mouse button instead of the right mouse button. Set it to

```
onlyOneMouseButton = <M2-Button-1>;<M2-B1-Motion>
```

Number of previous searches remembered As we have seen, with the 'back' button you can jump back to the previous search. This option controls the number of previous searches that are remembered. The default is 30, and if your machine has only a small amount of memory, you probably should not set it much higher, or Kombilo might run out of memory and crash. On the other hand, if you have lots of memory, it might be convenient to set it to a higher number, or even to 0, which means 'no limit': all searches are remembered, as long as there is enough memory.

Per-user configuration file

If in the *main* section, the `kombilo.cfg` file contains a configdir entry, like

```
configdir = ~
```

then this will be taken as a directory, and the `kombilo.cfg` file will be read from the `.kombilo` subdirectory of the configdir. In the configdir string, the tilde ~ will be replaced with the user's home directory (Linux). In this case, settings in the individual config file will overwrite those in the global file.

3.6.4 kombilo.app

You can change some 'global properties' like background color, type and size of the font used in the game list and in the text windows etc. by creating a file 'kombilo.app' in the main Kombilo directory. This is a plain text file; if you change it, please make sure to save the new version as plain text (ASCII), too.

Here is an example which shows the format of the file:

```
*font:                  Helvetica 10
*background:            grey88
*foreground:            black
*activeBackground:      grey77
*activeForeground:      black
*selectBackground:      grey77
*selectForeground:      black
*Listbox.background:    white
*Text.background:       white
*Entry.background:      white
*Canvas.background:     grey88
*Label.background:      grey88
```

Note: Changed in version 0.7.1: Before Version 0.7.1, the kombilo.app file was present by default. Before you create it, check whether you can obtain a look which is to your taste by *choosing a *theme** in the options menu.

3.6.5 Miscellaneous

The files containing the board image and the black and white stones are `icons/board.jpg`, `icons/black.gif` and `icons/white.gif`.

3.7 Troubleshooting

In case of errors, Kombilo writes some information to the file `kombilo.err` which is in the same directory as your *kombilo.cfg* file.

If you encounter problems, feel free to *contact me*.

3.8 Contributing

Kombilo intentionally is an open-source project. It has profited much from the contributions of its users in the past, and all your feedback and contributions are very much appreciated.

Development is concentrated on the Kombilo project page on BitBucket.

3.8.1 Tell me how you like Kombilo

Any kind of feedback is appreciated. Tell me which parts of Kombilo you like, and which ones need improvement. Did you use the Kombilo engine in your own scripts? I would be glad to learn about your results.

3.8.2 Ask questions, report bugs

If you have any problems, feel free to ask! Either by email at `ug@geometry.de`, or via the issue tracker.

3.8.3 Ideas

I have lots of ideas of new features I would like to implement, and I also would like to learn your ideas and priorities!

3.8.4 Development

If you have time to delve into Kombilo development, check out the mercurial repository:
`hg clone https://bitbucket.org/ugoertz/kombilo`

Feel free to fork the project and do send me pull requests for improvements or fixes you made.

3.8.5 Documentation

I try to maintain a reasonably complete documentation, but there surely are gaps and probably some inaccuracies. Please notify me, if you think that something is not explained well.

3.8.6 Windows/Mac OS X

I would love to add better support for Windows and/or Mac OS X users, however I do not have access to computers running either of these operating systems, right now. If you make progress on this, please tell me. I am also willing to discuss problems based on my experience with the previous Kombilo version for which I made a Windows installer.

3.9 Miscellaneous notes

3.9.1 References to commentaries

Kombilo has built in a list of references to game commentaries in the english go literature. The games are referenced by the Dyer signature (a signature assigned to the game which encodes the positions of move 20, 40, 60, 31, 51, 71, and which in practice characterizes a game uniquely); in particular Kombilo does not contain the game records. If Kombilo recognizes a game for which it has a reference, the corresponding line in the game list is highlighted by a light green background (by default - you can change this by editing the `kombilo.cfg` file), and a line which gives the actual reference is appended to the game info which is shown when that line in the game list is selected. (This is printed in blue, to show that it is not part of the game info proper, but was added by Kombilo.)

Currently, the list contains around 2000 references; in particular all issues of Go World, and most English books with game commentaries that I know of.

The references are stored in the file `references` in the `data` folder inside the main Kombilo directory. This is just a text file which you could edit yourself. The format should be self-explanatory. You can also download the current version of this file from the Kombilo source code repository and save it as the `references` file.

If you want only references to sources which you own to be shown, you can define exclude or include rules in the file `kombilo.cfg`.

Of course, additions to the list of references are very welcome. I think it would make sense to add references to other journals, like the American Go Journal, the British Go Journal, the Deutsche Go-Zeitung, the Revue Francaise de Go, etc.

3.9.2 Command line arguments

Kombilo.py

You can give file names of SGF files as command line arguments, and Kombilo will open these files upon startup. The file names should be given with the complete path. If blanks occur in the path or in the file name, it has to be put inside quotation marks.

v.py

The `v.py` SGF viewer accepts one SGF file name as the first argument, and optionally a move number as the second argument. The file will be opened at the specified move number.

3.9.3 Encodings

Kombilo can use SGF files with non-ASCII characters such as umlauts (äöü), accents (éèê), asian language characters, etc, **but currently it can only handle UTF-8-encoded files**. Of course, in addition the appropriate fonts to display these characters must be installed on your computer.

3.9.4 Requirements on SGF files

There are a few requirements on the SGF files that are used in the databases. They will be satisfied by ordinary game records, but might not be satisfied by "strange" SGF files.

First of all, the filename of an SGF file always has to end in '.sgf'.

In addition, at the very beginning an initial position can be set up. This is what happens in handicap games, for example. So handicap stones are treated correctly. It is also possible to set up an initial position consisting of black and white stones, like a go problem. On the other hand, "during the game", i.e. after the first black or white move has been played, no stones may be added or removed except for the ordinary alternating black/white moves (and except for captures, of course). In particular, all stones in the initial position have to be set up in the same node of the SGF file. Unfortunately, in a few handicap games of the Go Teaching Ladder, this is not the case; you will have to edit these files manually if you want to use them with Kombilo.

Empty nodes are skipped. When the usual 'black play' - 'white play' - 'black play' ... order is broken, Kombilo will stop processing the game in question at that point. This is another problem with games of the Go Teaching Ladder: in some of them, after a variation forked off a black/white move is not shown with the usual B/W tag, but with a AB/AW tag (which should be used to set up stones like handicap stones). Kombilo will process these games only until the first variation.

SGF collections: Kombilo's SGF editor can handle SGF files with several games in them, and so can the search engine. Nevertheless it is not a good idea to use games in that form, for performance reasons. It is better to split the collections, and then feed them into Kombilo. The problem with collections is that whenever the SGF file has to be read (for game info searches or to display the game info), the whole collection has to be read from disk, and has to be parsed.

The viewer does accept most SGF features, I think. In particular it handles variations (the navigation has to be done by clicking on the concerning points on the board), and adding/removal of stones during the game. It displays labels, but it does not properly display text labels with more than one letter/digit.

It ignores some of the new SGF tags like "good for black", "bad for white", ...

Kombilo ignores everything before the first '(;'. In particular, it will accept files with am email header and an SGF file after that. Be aware, though, that the header will be lost when you change the game info of that game: whenever Kombilo writes an SGF file, it will only write the game (resp. the game collection) itself.

3.9.5 Where to find game records

Here are some sources of game records:

- GoGoD encyclopedia has more than 70,000 games.
- Go4go has more than 28,000 games.
- Games of strong players on KGS
- List of links to SGF collections on u-go.net
- List of links to SGF collections on Sensei's library

CHAPTER

FOUR

USING THE KOMBILO ENGINE IN YOUR OWN SCRIPTS

This document describes how to use the Kombilo database from your own Python scripts, enabling you to do all kinds of data mining on your SGF databases.

4.1 Getting started

It is easiest to create some databases using Kombilo, and then just use the database files `kombilo.d*` (or first copy them somewhere).

Then, a pattern search can be done in a few lines:

```
# set up the KEngine, load the database files
K = KEngine()
K.gamelist.DBlist.append({ 'sgfpath': '.', 'name':('.', 'kombilo1'),
                           'data': None, 'disabled': 0})
K.loadDBs()

# let us check whether this worked
print K.gamelist.noOfGames(), 'games in database.'

# define a search pattern
p = Pattern('''
            .......
            .......
            ...X...
            ....X..
            ...OX..
            ...OO..
            .......
            ''', ptype=CORNER_NE_PATTERN, sizeX=7, sizeY=7)

# start pattern search
K.patternSearch(p)

# print some information
print K.patternSearchDetails()
```

For a slightly extended example, see `basic_pattern_search`. Instead of appending items to `K.gamelist.DBlist` manually as above, you can also use the py:meth:*GameList.populateDBs* method, see e.g. `sgftree`.

4.2 The scripts in the examples directory

4.2.1 basic_pattern_search.py

This script loads a Kombilo database, does a pattern search and a signature search and prints some information about the results after each search.

4.2.2 sgftree

This script takes an initial position, searches for it in the given database, and then searches for all continuations, then for all continuations in the newly found results etc. In this way, a tree of positions is computed, and in the end everything is written into an SGF file, with some information about the search results at each step.

Before starting the script, you need to write a configuration file. In the `[databases]` section, information about the databases to be used should be given, in the `[options]` section some options must be set.

Mandatory options are

```
output # name of the output file
initialposition # the initial position; see below for examples
depth # the highest move number that is considered
min_number_of_hits # variations with less hits are not considered
max_number_of_branches # if there are more continuations,
                       # only those with the most hits are considered
```

Further options:

```
gisearch # a query text for a game info search to be carried out
         # before the pattern searches
comment_head # text that should be prepended to every comment
```

The default value for `comment_head` is `@@monospace` which causes Kombilo to display the comment in a fixed width font. This is useful for output in tabular form.

In the `[searchoptions]` section, you can pass search options to Kombilo; possible choices are

```
fixedColor, nextMove, searchInVariations, moveLimit
```

Example config file (starting from the empty board):

```
[databases]
d0 = /home/ug/go/gogod10W, /home/ug/go/gogod10W, kombilo1
d1 = /home/ug/go/go4go, /home/ug/go/go4go, kombilo1
d2 = /home/ug/go/go4goN, /home/ug/go/go4goN, kombilo1
[options]
output = out1.sgf
# start with empty board:
initialposition = '(;)'
depth = 15
min_number_of_hits = 20
max_number_of_branches = 20
[searchoptions]
fixedColor = 1
```

Example config file (starting with opposing san ren sei):

```
[databases]
d0 = /home/ug/go/gogod11W, /home/ug/go/gogod11W, kombilo3
[options]
output = out2.sgf
initialposition = '(;AB[pd][pp][pj]AW[dd][dp][dj])'
depth = 15
```

```
min_number_of_hits = 5
max_number_of_branches = 20
[searchoptions]
fixedColor = 1
```

4.2.3 profiler

This scripts performs a number of pattern searches and writes a HTML file with information about the results and the time used for the searches. This makes it easy to compare Kombilo performance with different search parameters. Invoking the script for different versions of the underlying libkombilo library, you can also experiment with changes to the search algorithms, or compare new algorithms to the existing ones.

Usage:

Invoke the script as

```
./profiler.py s1
```

where `s1` is a subdirectory containing the following files.

Mandatory files:

```
kombilo1.d* # kombilo database files
hgsummary # a text file whose first line should contain information about the
          # revision (inside the hg source code repository) of libkombilo
          # used in this instance; to get started, just put the date (or
          # anything) as the first line of a text file.
jquery.js # The `JQuery <http://jquery.com>`_ javascript library which is
          # used in the HTML file produced by the script. Obtain a current
          # version from the `JQuery <http://jquery.com>`_ web site.
```

Optional files:

```
libkombilo.py, _libkombilo.so # the files providing the libkombilo library
                              # If you do not put them in the subdirectory,
                              # they are taken from the ``src/`` directory of
                              # your Kombilo installation.
```

Of course, you could easily to change the script to read the database from a different path or to use more than one database.

4.2.4 test_pattern_search

In this directory there are a couple of scripts which I used to test the pattern search for consistency. You can use them as starting points for your own scripts.

final_position.py

This script takes a database, and searches for the final position of each game in the database. If the number of results is different from 1, the file names of the games having this patterns are printed. (Typically these are games which have duplicates in the database, or which are very short.)

After searching for each final position, the script searches for the position at move 50 in each game.

Usage: invoke as

```
./various_tests.py s1
```

where `s1` is a subdirectory which contains data as for the `profiler` script. Output is to the console (instead of an HTML file).

various_tests.py

This script carries out a number of pattern searches, for various patterns and with various parameters, and checks the results for consistency.

Usage: invoke as

```
./various_tests.py s1
```

where `s1` is a subdirectory which contains data as for the `profiler` script, and to which the output html page is written.

4.3 API

4.3.1 The kombiloNG module

The kombiloNG module provides much of the Kombilo functionality without the Graphical User Interface. You can use it to do pattern searches etc. in your Python scripts.

class `kombiloNG.`**`Cursor`**(**args*, ***kwargs*)
A Cursor which is used to traverse an SGF file. See the documentation of the `sgf` module for further details.

class `kombiloNG.`**`GameList`**
A Kombilo list of games. The list can consist of several Kombilo databases. You do not construct instances of this class yourself. Rather, every `KEngine` instance `K` has a unique instance `K.gamelist` of `GameList`.

As in Kombilo, the GameList maintains a list of games that are "currently visible" (think of all games matching some pattern). All search methods and many other methods work with this "current list".

`addTag`(*tag*, *index*)
Set tag on game at position index in the current list.

`exportTags`(*filename*, *which_tags*=[ ])
Export all tags in all non-disabled databases into the file specified by `filename`.

If which_tags is specified, then it has to be a list of positive integers, and only the tags in the list are exported.

`getIndex`(*i*)
Returns dbIndex, j, such that self.DBlist[dbIndex]['current'][j] corresponds to the i-th entry of the current list of games.

`getProperty`(*index*, *prop*)
Return a property of the game at position `index` in the current list of games. Here `prop` should be one of the following constants:

- `GL_FILENAME` - the filename
- `GL_PB` - the black player
- `GL_PW` - the white player
- `GL_RESULT` - the result
- `GL_SIGNATURE` - the symmetrized Dyer signature
- `GL_DATE` - the date.

`getSGF`(*index*)
Return the SGF source of the game at position `index` in the current list of games.

getTags(*index*)
Get all tags of the game at position index in the current list.

get_data(*i*, *showTags=True*)
Return entry in line i of current list of games (as it appears in the Kombilo game list window).

importTags(*filename*)
The file given by filename should be a file to which previously tags have been exported using `exportTags()`.

This method imports all the tags into the current databases. The games are identified by the Dyer signature together with a hash value of their final position. So unless there are duplicates in the database, this should put the tags on those games where they were before exporting. In case of duplicates, all duplicates will receive the corresponding tags.

listOfCurrentSGFFiles()
Return a list of file names for all SGF files of games in the current list of games.

noOfGames()
Return the number of games in the current list of games.

noOfHits()
Return the number of hits for the last pattern search.

noOfSwitched()
Return the number of hits where the colors are reversed for the last pattern search.

populateDBlist(*d*)
Add the databases specified in the dictionary `d` to this GameList. `d` must have the following format:

For each key `k`, `d[k]` is a list of three entries. The first entry is the `sgfpath`, i.e. the path where the SGF files of this database are stored. The second entry is the path where the Kombilo database files are stored, and the third entry is the name of these database files, without the extension.

The keys are assumed to be strings. If''k'' ends with 'disabled', then the disabled flag will be set for the corresponding database.

After adding the databases in this way, you must call `KEngine.loadDBs()` to load the database files.

printGameInfo(*index*)
Return a pair whose first entry is a string containing the game info for the game at index. The second entry is a string giving the reference to commentaries in the literature, if available.

printSignature(*index*)
Return the symmetrized Dyer signature of the game at `index` in the current list of games.

reset()
Reset the list, s.t. it includes all the games from self.data.

class kombiloNG.**KEngine**
This is the class which you use to use the Kombilo search functionality.

After instantiating it, you need to tell the gamelist which databases you want to use, e.g. using `GameList.populateDBlist()`, and then call `loadDBs()`. Afterwards you can use `patternSearch()`, for instance.

See the Kombilo documentation on further information how to get started.

Further notes.

After a pattern search, the continuations are assembled into the list `self.continuations`, whose entries are lists [total number of hits, x-coordinate in currentSearchPattern, y-coordinate in currentSearchPattern, number of black continuations, number of black wins after black play here, number of black losses after black play here,

number of black plays here after tenuki, number of white continuations, number of black wins after white play here, number of black losses after white play here, number of white plays here after tenuki, label used on the board at this point]

addDB(*dbp*, *datap=('', '#')*, *recursive=True*, *filenames='*.sgf'*, *acceptDupl=True*, *strictDuplCheck=True*, *tagAsPro=0*, *processVariations=1*, *algos=None*, *messages=None*, *progBar=None*, *showwarning=None*, *index=None*)

Call this method to newly add a database of SGF files.

Parameters:

- dbp: the path where the sgf files are to be found.
- datap: the path where the database files will be stored. Leaving the default value means: store database at dbp, with base filename 'kombilo'. Instead, you can specify a pair (path, filename). Then path/filenameN.d? will be the locations of the database files. Every Kombilo database consists of several files; they will have names with ? equal to a, b, ... N is a natural number chosen to make the file name unique.
- recursive: specifies whether subdirectories should be included recursively
- messages: a 'message text window' which receives status messages
- progBar: a progress bar
- showwarning: a method which display warnings (like Tkinter showwarning)

copyCurrentGamesToFolder(*dir*)

Copy all SGF files belonging to games in the current list to the folder given as `dir`.

dateProfile(*intervals=None*)

Return the absolute numbers of games in the given date intervals among the games in the current list of games.

Default value for `intervals` is

```
[ (0, 1900), (1900, 1950), (1950, 1975), (1975, 1985), (1985, 1992),
(1992, 1997), (1997, 2002), (2002, 2006), (2006, 2009), (2009, 2013),
]
```

dateProfileRelative()

Return the ratios of games in the current list versus games in the whole database, for each of the date intervals specified in `dateProfile()`.

gameinfoSearch(*query*)

Do a game info search on the current list of games.

- `query` provides the query as part of an SQL clause which can be used as an SQL WHERE clause. Examples:

```
date >= '2000-03-00'
PB = 'Cho Chikun'
PB like 'Cho%'
PW like 'Go Seigen' and not PB like 'Hashimoto%'
```

After the `like` operator, you can use the percent sign `%` as a wildcard to mach arbitrary text.

The columns in the database are

```
PB (player black)
PW (player white)
RE (result)
EV (event)
DT (the date as given in the sgf file)
date (the date in the form YYYY-MM,DD)
```

```
filename
sgf (the full SFG source).
```

gameinfoSearchNC(*query*)
Returns the number of games matching the given query (see `gameinfoSearch()` for the format of the query) **without changing the list of current games**.

loadDBs(*progBar=None*, *showwarning=None*)
Load the database files for all databases that were added to the gamelist.

parseReferencesFile(*datafile*, *options=None*)
Parse a file with references to commentaries in the literature. See the file `src/data/references` for the file format.

The method builds up `self.gamelist.references`, a dictionary which for each Dyer signature has a list of all references for the corresponding game.

patternSearch(*CSP*, *SO=None*, *CL='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz123456789'*, *FL={}*, *progBar=None*)
Start a pattern search on the current game list.

- CSP must be an instance of `Pattern` - it is the pattern that is searched for.
- You can specify search options as *SO* - this must be an instance of `lk.SearchOptions` (see below).
- `CL`, `FL`, `progBar` are used with the Kombilo GUI.

Search options. Create an instance of `lk.SearchOptions` by

```
so = lk.SearchOptions
```

You can then set particular options on `so`, e.g.:

```
so.fixedColor = 1
so.searchInVariations = false
```

Available options:

- fixedColor, values: 0 = also search for pattern with colors reversed; 1 = fix colors as given in pattern; default value is 0
- nextMove, values: 0 either player moves next, 1 = next move must be black, 2 = next move must be white; default value is 0
- moveLimit, positive integer; pattern must occur at this move in the game or earlier; default value is 10000
- trustHashFull, boolean, values: true = do not use ALGO_MOVELIST to confirm a hit given by ALGO_HASH_FULL, false = use ALGO_MOVELIST to confirm it; default value is false
- searchInVariations, boolean; default value is true
- algos, an integer which specifies which algorithms should be used; in practice, use one of the following:

```
lk.ALGO_FINALPOS | lk.ALGO_MOVELIST
lk.ALGO_FINALPOS | lk.ALGO_MOVELIST | lk.ALGO_HASH_FULL
lk.ALGO_FINALPOS | lk.ALGO_MOVELIST | lk.ALGO_HASH_FULL | lk.ALGO_HASH_CORNER
```

  The default is to use all available algorithms.

patternSearchDetails(*exportMode='ascii'*, *showAllCont=False*)
Returns a string with information on the most recent pattern search.

signatureSearch(*sig*)
Do a signature search for the Dyer signature `sig`.

tagSearch(*tag*)
Do a tag search on the current game list.

tag can be an expression like `H and (X or not M)`, where H, X, M are abbreviations for tags (i.e. keys in self.gamelist.customTags). In the simplest example, tag == `H`, i.e. we just search for all games tagged with `H`.

class kombiloNG.**Node**(**args*, ***kwargs*)
A Node of an SGF file. Also see the documentation of the `sgf` module.

exportPattern(*boardsize=19*)
Return a full board pattern with the position at this node.

class kombiloNG.**Pattern**(*p*, ***kwargs*)
A pattern, i.e., a configuration of black and white stones (and empty spots, and possibly wildcards) on a portion of the go board.

To create a pattern, pass the following arguments to Pattern:

- p: The pattern as a string (`...XXO..X`). Blanks and line breaks will be ignored. Commas (to mark hoshis) will be replaces by periods.
- ptype (optional): one of

```
CORNER_NW_PATTERN, CORNER_NE_PATTERN, CORNER_SW_PATTERN, CORNER_SE_PATTERN
# fixed in specified corner

SIDE_N_PATTERN, SIDE_W_PATTERN, SIDE_E_PATTERN, SIDE_S_PATTERN
# slides along specified side

CENTER_PATTERN
# movable in center

FULLBOARD_PATTERN.
```

- sizeX, sizeY: the size (horizontal/vertical) of the pattern (not needed, if ptype is `FULLBOARD_PATTERN`).
- anchors (optional): A tuple (right, left, top, bottom) which describe the rectangle containing all permissible positions for the top left corner of the pattern.

One of ptype and anchors must be present. If ptype is given, then anchors will be ignored.

- contlist (optional): A list of continuations, in *SGF format*, e.g. `;B[qq];W[de];B[gf]`,
- topleft (optional): a pair of coordinates, specifying the top left corner of the pattern, needed for translating contlist into coordinates relative to the pattern
- contsinpattern (optional; used only if contlist is not given): `X` (black) or `O` (white). If given, the labels 1, 2, 3, ... in the pattern are extracted and handled as continuations, with 1 played by the specified color.
- contLabels (optional): A string of same size as p, with labels that should be used for labelling continuations.

> **Warning:** Continuation and captures
> With the `Pattern` class it is not currently possible to deal with captures made by one of the moves of the continuation list. While the libkombilo library allows to do this, I have yet to think of a good interface to access this functionality.

getInitialPosAsList(*hoshi=False*, *boundary=False*)
Export current pattern as list of lists, like [['.', 'X', '.'], ['O', '.', '.']]

If boundary==True, a boundary of spaces, '-', 'l', '+'s is added. If hoshi==True, hoshi points are marked with ','. (Of course, this is only applicable for fullboard or corner patterns, or patterns with fixed anchor.)

kombiloNG.**symmetrizeSig**(*s*)
Given a signature s, compute the 'rotated'/'mirrored' signatures, and return the first, w.r.t. lexicographic order, of all these. Games which differ only by a symmetry of the board will thus have the same symmetrized signature.

4.3.2 The sgf module

The sgf module provides functionality for handling SGF files.

class sgf.**Cursor**(*sgf*, *sloppy=False*, *encoding='utf8'*)
The Cursor class takes SGF data (as a string) and provides methods to traverse the game and to retrieve the information for each node stored in the SGF file.

To create a Cursor instance, call Cursor with the following arguments:

- sgf: The SGF data as a string.
- sloppy (optional, default is False): If this is True, then the parser tries to ignore deviations from the SGF format.
- encoding (optional, default is 'utf-8'): **This option is currently not used.** Later, the parser will decode the file with the specified encoding.

currentNode()
Get an instance of class Node for the node the cursor currently points to.

exportGame(*gameNumber=None*)
Return a string with the game attached to self in SGF format (with character encoding utf-8!). Depending on gameNumber:

- if None: only the currently "active" game in the collection is written (as specified by self.currentGame)
- if an integer: the game specified by gameNumber is written
- it a tuple of integer: the games for this tuple are written
- if == 'ALL': all games are written

getRootNode(*n*)
Get the first node of the n-th node of this SGF game collection. Typically, SGF files contain only a single game; getRootNode(0) will give you its root node.

next(*n=0*)
Go to n-th child of current node. Default for n is 0, so if there are no variations, you can traverse the game by repeatedly calling next().

noChildren()
Returns the number of children of the current node, i.e. the number of variations starting here.

previous()
Go to the previous node.

updateRootNode(*data*, *n=0*)
Update the root node of the n-th game in this collection.

data is a dictionary which maps SGF properties like PB, PW, ... to their values.

class sgf.**Node**(*node*)
The Node class represents a single node in a game. This class is a wrapper for lk.Node class. It has dictionary style access to sgf property values.

This class does not inherit from lk.Node. To construct a Node, pass an lk.Node instance to __init__. It is stored as self.n.

You can check whether a Node `node` has an SGF property and retrieve its value like this: `if 'B' in node: print node['B']`. Similarly, using `node['B'] = ('pp', )` and `del node['B']` you can set values and delete properties from `node`.

add_property_value(*ID*, *item*)
Add `item` to the list `self[ID]`.

get_move_number()
Returns the move number where the node sits inside the game. This is a list of non-negative integers. The entries with even indices mean "go right by this amount in the game tree"; the entries at odd places mean "go down as many steps as indicated, i.e. pass to the corresponding sibling".

pathToNode()
Returns 'path' to the specified node in the following format: `[0,1,0,2,0]` means: from rootNode, go * to next move (0-th variation), then (* to first variation, then * to next move (0-th var.), then * to second variation, * then to next move.

In other words, a cursor c pointing to rootNode can be moved to n by `for i in n.pathToNode(): c.next(i)`.

remove(*ID*, *item*)
Remove `item` from the list `self.n[ID]`.

4.3.3 Libkombilo constants

Pattern types:

```
CORNER_NW_PATTERN
CORNER_NE_PATTERN
CORNER_SW_PATTERN
CORNER_SE_PATTERN

SIDE_N_PATTERN
SIDE_W_PATTERN
SIDE_E_PATTERN
SIDE_S_PATTERN

CENTER_PATTERN

FULLBOARD_PATTERN
```

Algorithms:

```
ALGO_FINALPOS
ALGO_MOVELIST
ALGO_HASH_CORNER
ALGO_HASH_FULL
```

4.4 Libkombilo

To study the underlying C++ library in detail, look at the Libkombilo documentation. A good starting point is the `cpptest.cpp` program in `lk/examples`.

CHAPTER

FIVE

HISTORY/UPGRADING

5.1 Upgrading

When upgrading between major versions (e.g., 0.5 to 0.7), you always have to newly process your databases. Since processing is pretty fast by now, it should not hurt too much.

Between minor versions in the 0.7 branch, the database file format did not change. You can copy your kombilo.cfg file (which contains, among many options, the database locations) to the new directory, and thus keep your databases, including all tags.

5.2 History/Change log

5.2.1 0.7

0.7.4 (August 2012)

Fixed bug in search algorithm. Thanks to John Fairbairn for pointing it out.

0.7.3 (May 2012)

- Add *Find duplicates* method.
- Fix `goto move` method (Shift + Right click).
- `.sgf` as default extension upon saving files.

0.7.2 (April 2012)

Bug fixes. Added mouse wheel support, and - on Linux - support for back/next mouse buttons. Option to always maximize window (Windows only). Updated references to commentaries.

Note: Update from version 0.7 or 0.7.1 to 0.7.2:

While reprocessing your databases is not necessary for Kombilo to be working correctly, it will greaty speed up the date profile computation, so it is advisable to do this.

0.7.1 (April 2012)

Major changes:

- provide a Windows installer.

Selected minor fixes:

- Make reprocess export/import tags automatically.
- Optionally put search history as a tab in notebook in right column. Thanks to crux@lifein19x19 for the suggestion.
- Pass focus to boardFrame upon double click in game list. Thanks to crux@lifein19x19 for the suggestion.
- Update references to commentaries.
- optionally Kombilo avoids the use of Python Imaging Library (PIL). With this option, Kombilo can be made to work on Mac OS X. Thanks to RBerenguel@lifein19x19 for testing things on Mac OS.
- Fixed some issues regarding the handling of bad SGF files.

0.7 (March 2012)

Version 0.7 brought lots of changes to Kombilo and Libkombilo, including

Kombilo

- Kombilo now makes use of the libkombilo C++ library; it can search in variations, and search for move sequences
- Tags for games
- Show *date profile* for current list of games
- Much better support for using Kombilo in your own Python scripts; several example scripts included
- Update the GUI: nicer icons, keyboard bindings, better handling of large game lists, etc.

Libkombilo

- Use multiple processor cores/processors, if available
- Allow tagging of games
- Change the data file format the search algorithms use to make startup and searching faster
- Many bug fixes and small improvements

Furthermore, there have been a number of changes to make the development process easier and more pleasant, e.g.

- switch to Mercurial as the versioning system for the source code, and host the project on BitBucket,
- use Sphinx for producing the documentation
- use Fabric to make deployment easier

5.2.2 "0.6": Libkombilo (2006)

There was no *Kombilo 0.6* release, but in 2006 I partially rewrote the pattern matching algorithms, and isolated them from the graphical user interface. This made it easy to include it as a library into other programs. Since version 3.0 (2007) it is included in Drago.

The libkombilo library can ...

- search for corner patterns, full board patterns, and patterns anywhere on the board, of course taking into account symmetries (rotation, mirroring), and -unless switched of- color reversal.
- handle for any (square) board size.
- search for continuations, i.e. you give an initial pattern (possibly empty), and then a sequence of moves which have to occur in every hit in the given order.
- search in games with variations, and find results within variations as well.

5.2.3 0.5 (2002-2004)

- Kombilo comes with a complete SGF editor: so you can add variations of your own, comment the game, add labels etc. The SGF editor can also handle collections, i.e. SGF files containing several games. The tree structure of the current game is shown in a separate window. You can rotate/mirror SGF files.
- Kombilo now comes with a built in list of references to commentaries of games in the English go literature. (NB: Kombilo does not come with the game records, but recognizes the games by the Dyer signature.) Those games in your database which Kombilo finds in its list are marked in the game list, and in the game info a reference to the journal/book which has the commentary is given. Currently the list contains around 1200 references, and includes references to the game commentaries in 85 issues of Go World and in most English go books with game comentaries.
- The previous search patterns are now shown on small boards in a scrollable separate window. Thus you can switch back and forth between different search patterns much more easily. This also works much better now with different SGF files. In particular, you can load games from the game list directly to the Kombilo main board, and then search for patterns which arise in that game.
- You can sort the game list with respect to one of several criteria (besides the default, sort by filename, you can now also sort by date, white player or black player). You can also easily change the order of the databases.
- You can refine pattern searches by filtering who moves next in the search pattern.

5.2.4 0.4 (2002)

- Custom menus: menus which you can edit yourself. Upon selecting a menu entry, the following actions can be performed: search for a predefined pattern; search for predefined game information (player, event, ...); open the web browser with some html file. Thus you could create a “Fuseki/Joseki pattern” menu, a “Players” or a “Titles” menu.
- Even faster SGF parser. (On my computer, Kogo’s joseki dictionary now comes up immediately.)
- Better handling of large databases.
- First SGF editing features: you can now edit the game information, and the comments. (Make sure to have backups of important files ;-))
- Optionally include the whole game list when exporting search results.
- Indicate color swap in the list of results
- Searches with lots of matches are considerably faster now.

5.2.5 0.3 (2002)

- The search engine has been partially rewritten; in particular two subtle bugs have been fixed. The use of hash tables makes joseki and fuseki searches considerably faster.

- A faster SGF parser. With the new parser, Kogo's joseki dictionary, a huge file, can be read in in a few seconds, and thus can be conveniently used with Kombilo to study Joseki.
- Winning percentages for continuations; show how often some continuation is played after tenuki.
- Export function for search results (either as plain text, or in a format suitable for use in Sensei's Library)

5.2.6 0.2 (2002)

- More comfortable game info search (time period, players, event, ...)
- 'Back' button to return to the previous search.
- More convenient user interface. In particular, the two windows will fit on your screen (800x600 or bigger) without overlapping now.
- Display Black/White winning percentages. More detailed statistics on the continuations in a search pattern.
- Check for duplicates in the data base (with the Dyer signature), search games by signature.

5.2.7 0.1 (October 2001)

The first Kombilo version. It already had the basic pattern search functionality (including the C++ extension), but was still rough around the edges.

CHAPTER

SIX

LICENSE

6.1 License for the Kombilo source code

```
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies
of the Software, and to permit persons to whom the Software is furnished to do
so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

(This is the so-called MIT License (or X11 License), see http://www.opensource.org/licenses/mit-license .)

6.2 Icons

Many of the icons are taken from the Tango icon collection, by Jones Lee. See http://art.gnome.org/themes/icon/1150 They are published under the CreativCommons Attribution-Share alike License (CC BY-SA 2.0), see http://creativecommons.org/licenses/by-sa/2.0/

I slightly modified some of the Tango icons (media-playback-back-K.png, media-playback-pause-K.png, media-playback-start-K.png) and publish these derived works under the CC BY-SA 2.0 license.

The other icons (w.gif, b.gif, wb.gif, bw.gif, 123.png, abc-l.png, abc-u.png, tr.gif, sq.gif) are dual-licensed under the CC BY-SA 3.0 license (http://creativecommons.org/licenses/by-sa/3.0/) and the MIT license.

6.3 Board, stone images

The images of the go board and the back and white stone were contributed by Patrice Fontaine.

6.4 Logo

The u-go.net logo is (C) Ulrich Goertz; it may be freely distributed together with the Kombilo documentation, but may not be used in other contexts, and may not be altered.

6.5 Tooltip

The code for the tooltip (in src/tooltip/tooltip.py) was contributed by Tucker Beck as recipe 576688 to the ActiveState community under the MIT license, see http://code.activestate.com/recipes/576688-tooltip-for-tkinter/ and slightly adapted for Kombilo.

6.6 Windows installer

The Windows installer bundles several programs and libraries in addition to Kombilo itself:

6.6.1 Python

```
PSF LICENSE AGREEMENT FOR PYTHON 2.7.2

1. This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"),
and the Individual or Organization ("Licensee") accessing and otherwise using
Python 2.7.2 software in source or binary form and its associated
documentation.

2. Subject to the terms and conditions of this License Agreement, PSF hereby
grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce,
analyze, test, perform and/or display publicly, prepare derivative works,
distribute, and otherwise use Python 2.7.2 alone or in any derivative version,
provided, however, that PSF's License Agreement and PSF's notice of copyright,
i.e., "Copyright © 2001-2012 Python Software Foundation; All Rights Reserved"
are retained in Python 2.7.2 alone or in any derivative version prepared by
Licensee.

3. In the event Licensee prepares a derivative work that is based on or
incorporates Python 2.7.2 or any part thereof, and wants to make the derivative
work available to others as provided herein, then Licensee hereby agrees to
include in any such work a brief summary of the changes made to Python 2.7.2.

4. PSF is making Python 2.7.2 available to Licensee on an "AS IS" basis. PSF
MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE,
BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY
OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF
PYTHON 2.7.2 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 2.7.2 FOR
ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF
MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 2.7.2, OR ANY DERIVATIVE
THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material breach
of its terms and conditions.
```

```
7. Nothing in this License Agreement shall be deemed to create any relationship
of agency, partnership, or joint venture between PSF and Licensee. This License
Agreement does not grant permission to use PSF trademarks or trade name in a
trademark sense to endorse or promote products or services of Licensee, or any
third party.

8. By copying, installing or otherwise using Python 2.7.2, Licensee agrees to
be bound by the terms and conditions of this License Agreement.
```

Link: Python License.

6.6.2 Python Imaging Library PIL

```
The Python Imaging Library (PIL) is

Copyright © 1997-2011 by Secret Labs AB
Copyright © 1995-2011 by Fredrik Lundh

By obtaining, using, and/or copying this software and/or its associated
documentation, you agree that you have read, understood, and will comply
with the following terms and conditions:

Permission to use, copy, modify, and distribute this software and its
associated documentation for any purpose and without fee is hereby
granted, provided that the above copyright notice appears in all copies,
and that both that copyright notice and this permission notice appear in
supporting documentation, and that the name of Secret Labs AB or the
author not be used in advertising or publicity pertaining to
distribution of the software without specific, written prior permission.

SECRET LABS AB AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO
THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND
FITNESS. IN NO EVENT SHALL SECRET LABS AB OR THE AUTHOR BE LIABLE FOR
ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER
RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF
CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN
CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
```

Link: PIL license.

6.6.3 Python MegaWidgets

```
Copyright 1997-1999 Telstra Corporation Limited, Australia Copyright 2000-2002
Really Good Software Pty Ltd, Australia

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so,
subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
```

```
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

Link: Python MegaWidgets license.

6.6.4 ConfigObj

```
Copyright (c) 2004 - 2010, Michael Foord & Nicola Larosa

All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:


* Redistributions of source code must retain the above copyright
  notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice,
  this list of conditions and the following disclaimer in the documentation
  and/or other materials provided with the distribution.

* Neither the name of Michael Foord nor Nicola Larosa may be used to
  endorse or promote products derived from this software without specific prior
  written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS IMPLIED
WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT ( OTHERWISE) OUT OF
THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

Link: ConfigObj license

6.6.5 SQLite

```
All of the deliverable code in SQLite has been dedicated to the public domain
by the authors. All code authors, and representatives of the companies they
work for, have signed affidavits dedicating their contributions to the public
domain and originals of those signed affidavits are stored in a firesafe at the
main offices of Hwaci. Anyone is free to copy, modify, publish, use, compile,
sell, or distribute the original SQLite code, either in source code form or as
a compiled binary, for any purpose, commercial or non-commercial, and by any
means.
```

Link: SQLite license

6.6.6 Boost

Most Boost libraries comply with the license below. While for some, other licenses might apply, they are also freely distributable (and, since the Boost source code is not included in the Kombilo installer, it is not even required that the

license be stated here). For details, see the link below.

```
Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization
obtaining a copy of the software and accompanying documentation covered by
this license (the "Software") to use, reproduce, display, distribute,
execute, and transmit the Software, and to prepare derivative works of the
Software, and to permit third-parties to whom the Software is furnished to
do so, all subject to the following:

The copyright notices in the Software and this entire statement, including
the above license grant, this restriction and the following disclaimer,
must be included in all copies of the Software, in whole or in part, and
all derivative works of the Software, unless such copies or derivative
works are solely in the form of machine-executable object code generated by
a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT
SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE
FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
DEALINGS IN THE SOFTWARE.
```

Link: Boost library license.

6.6.7 Further acknowledgments

The libkombilo library is compiled using the MinGW compiler, an exe file is produced using py2exe, the installer is made with InnoSetup, and the documentation is compiled by the Sphinx package. Fabric eases the deployment process. VirtualBox turned out to be magnificent for simultaneously developing on Linux and Windows.

6.7 Acknowledgments

Over the years, many people directly or indirectly supported the development of Kombilo by offering code, general feedback, suggestions or bug reports, in particular: Gilles Arcas, Daniel Balsom, Arend Bayer, Ruben Berenguel, Simon Cozens, Sergey Datskovskiy, Fabrice de Volder, Jon Diamond, John Fairbairn, Patrice Fontaine, Christian Gawron, Sorin Gherman, Daniel Gilder, Steffen Glueckselig, Igor Goliney, Rene Grothmann, Alberto Hernando, Anders Kierulf, Tobias Klaus, Mace Lee, Marc A. Lehmann, Andre Prasetya, Alberto F. Rezza, Hendrik Reinke, Uwe Richter, Douglas Ridgway, Jan van Rongen, Bernhard Runge, Thomas Schmid, Thomas Schmid-Lindner, Bernd Schmidt, David Sigaty, Falko Spiller, Neil Stevens, Fred Strauss, Dan Stromberg, Jean-Pierre Vesinet, Christian Wenzel.

PYTHON MODULE INDEX

INDEX