cpe — Chess-Problem-Editor

Stefan Höning
Neuss

März 2015

Vor über 10 Jahren habe ich — damals noch unter DOS — ein Programm entwickelt, mit dem man visuell die Positionen von Schachdiagrammen zur weiteren Bearbeitung mit LaTeX und dem `diagram.sty` erfassen konnte. Nachdem ich im letzten Jahr die Programmiersprache Python erlernt habe, schien mir diese geeignet einen neuen Anlauf zu starten eine Software zu entwickeln, die alle Möglichkeiten des *diagram.sty* unterstützt. Dieses Dokument beschreibt die Bedienung und in Abschnitt 6 die Installation dieser Software.

1 Einleitung

Die Software besteht aus mehreren Teilen, die in den folgenden Abschnitten beschrieben sind. Das *Haupt-Programm* `cpe.py` dient der Erfassung der Diagramme. Daneben gibt es ein Programm zur (manuellen) Pflege der Autoren und Quellen `cpdb.py` sowie verschiedene Scripten, die bestimmte Aufgaben automatisieren oder zumindest unterstützen.

2 Die Erfassung von Schachproblemenen

Abbildung 1 zeigt das Hauptfenster des *ChessProblemEditor*s. Im linken Teil der Anwendung sind die visuellen Eingabeelemente für die Stellung enthalten. Im rechten Teil können alle anderen Informationen erfasst werden.

Über das `File` Menu können Dateien geöffnet und gespeichert, sowie das Programm beendet werden. Über das `Problems` Menu kann innerhalb der Probleme in der Datei navigiert werden. Für alle diese Aktionen kann man auch die entsprechenden Keyboard-Shortcuts verwenden, die in Tabelle 1 angegeben sind. Im `Compile` Menu kann das aktuelle Document mit LaTeX übersetzt und angezeigt werden.

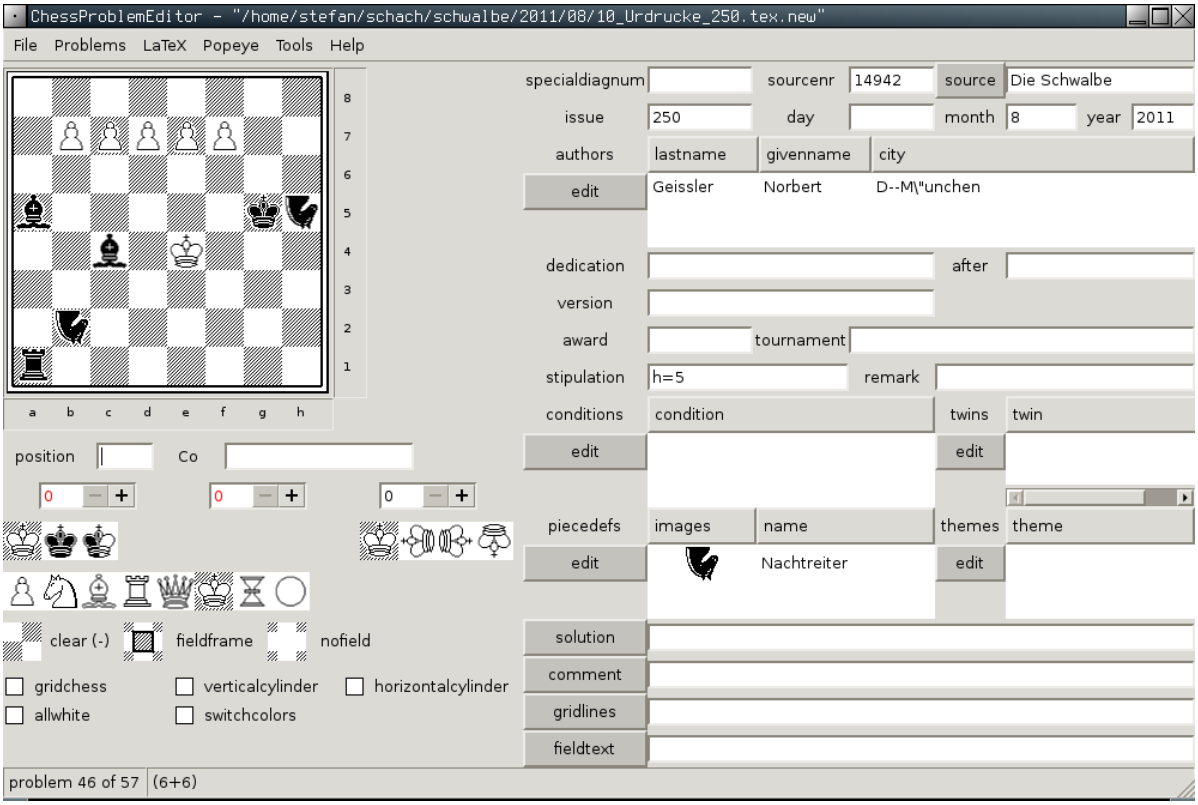

Abbildung 1: Das Haupt-Fenster des *ChessProblemEditor*s.

File-Menu	**Shortcut**	Problems-Menu	Shortcut
New	Ctrl-N	First	Home
Open	Ctrl-O	Previous	Pg-Up
Save	Ctrl-S	Next	Pg-Down
Save as	Ctrl-A	Last	End
Quit	Ctrl-Q	Insert	Shift-Ins
		Append	Shift-Enter
		Delete	Shift-Delete
		Change board size	Ctrl-B

Tabelle 1: Keyboard Shortcuts für die Menu-Aktionen

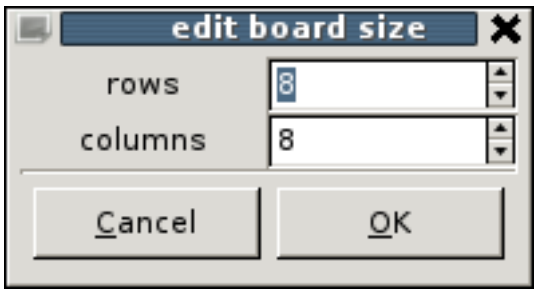

Abbildung 2: Der Dialog zur Einstellung der Brettgröße.

2.1 Erfassung der Stellung

Zur Erfassung der Stellung kann entweder mit der Maus gearbeitet werden, oder mit dem *position* Textfeld die Tastatur genutzt werden.

Bei Eingabe per Tastatur mit der *Leertaste* zwischen den Farben gewechselt werden — die Reihenfolge ist ***Weiss–Schwarz–Neutral***. Im Übrigen ist die Eingabe wie man sie auch diktieren würde: Figuren-Typ (K – König, D – Dame, T – Turm, L – Läufer, S – Springer, B – Bauer), optional die Drehung der Figur (l – links, r – rechts und u – auf dem Kopf stehend) gefolgt von einer Liste der Felder. Die Position in Abbildung 1 kann über folgende Eingabe erzielt werden:

```
ke4bb7c7d7e7f7 kg5ta1la5c4sub2h5
```

Bei Fehleingaben kann man durch Eingabe eines *minus* – in den Löschmodus schalten. Alternativ kann man mit der Maus das *clear* Feld unter der Figurenauswahl betätigen. Diesen verlässt man wieder durch Eingabe eines Figurentyps und — oder natürlich durch Auswahl einer Figur mit der Maus.

Auch größere und kleinere Schachbretter können erfasst werden. Mittels des Menu-Punkts *Problems / Change board size* oder `Ctrl-B` kann über den in Abbildung 2 dargestellten Dialog die Brettgröße geändert werden. Bei grösseren Brettern wird automatisch eine Scrollbar angezeigt.

WARNUNG: Bei größeren und kleineren Brettern funktioniert die Erfassung über die Tastatur nur bedingt.

2.2 Erfassung der übrigen Problem Informationen

Neben dem Eingabefeld für die Position kann noch die Information zur Computerprüfung erfasst werden.

Im rechten Teil des Fensters kann man die übrigen Information erfassen — für die im `diagram.sty` entsprechende Kommandos zur Verfügung stehen. Dabei sind die folgenden einfache Texteingabe-Felder: `specialdiagnum`, `sourcenr`, `issue`, `day`, `month`, `year`, `dedication`, `after`, `award`, `tournament`, `stipulation`, `remark`.

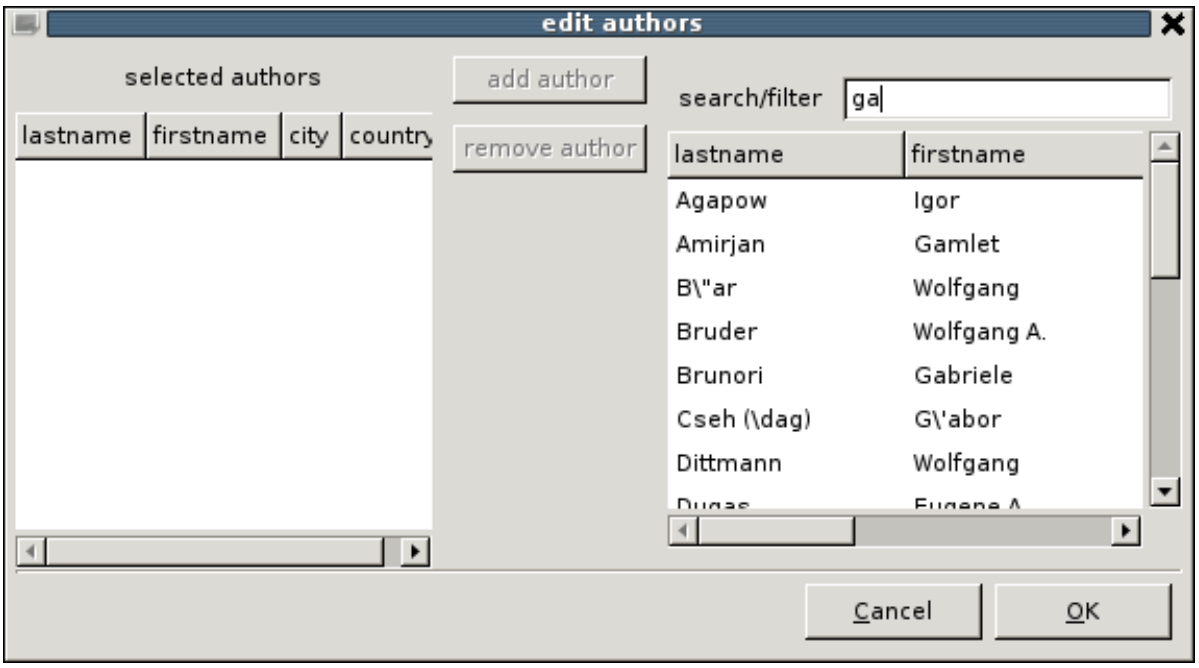

Abbildung 3: Die Auswahl von Autoren

Bei `source` muss man den Button betätigen, um den Namen aus der Liste der in der Datenbank gespeicherten Quelle auszuwählen.

Bei `authors` öffnet man durch Auswahl des Buttons `edit` den in Abbildung 3 dargestellten Dialog zur Auswahl der Autoren. In der rechten Listbox werden alle Autoren angezeigt, die das Suchkriterium (`search/filter`) enthalten. In der rechten Listbox selektierte Autoren kann man mittels des `add author` Buttons zum Problem hinzufügen.

Bei `conditions` öffnet man durch Auswahl den Buttons `edit` den in Abbildung 4 dargestellten Dialog zur Auswahl der Bedingungen. Der Dialog wird im Prinzip genauso bedient, wie der Dialog zur Auswahl der Autoren. Ist eine Bedingung nicht vorhanden bzw. benötigt spezielle Parameter, so kann die Bedingung durch Auswahl des Buttons `add other condition` in einem weiteren Dialog erfasst werden, der in Abbildung 5 dargestellt ist.

Bei `piecedefs` öffnet man durch Betätigen des Buttons `edit` den in Abbildung 6 dargestellten Dialog zur Festlegung der Märchenfiguren. Durch Betätigen des Buttons `scan problem` kann die Liste auf der linken Seite mit den im Diagramm vorkommenden Symbolen von Märchenfiguren gefüllt werden. Nach Auswahl einer Zeile innerhalb dieser Liste der Symbole und des Namens einer Märchenfigur kann man diese mittels des `set` Buttons einander zuordnen.

Bei `solution`, `comment`, `gridlines` und `fieldtext` wird in dem Textfeld jeweils die erste Zeile angezeigt. Durch Betätigen der jeweiligen Schaltfläche öffnet sich der in Abbildung 7 dargestellte Dialog.

Bei `conditions` und `twins` öffnet man über den Button `edit` einen Dialog, über den man die Liste der Bedingungen bzw. Zwillinge Pflegen kann.

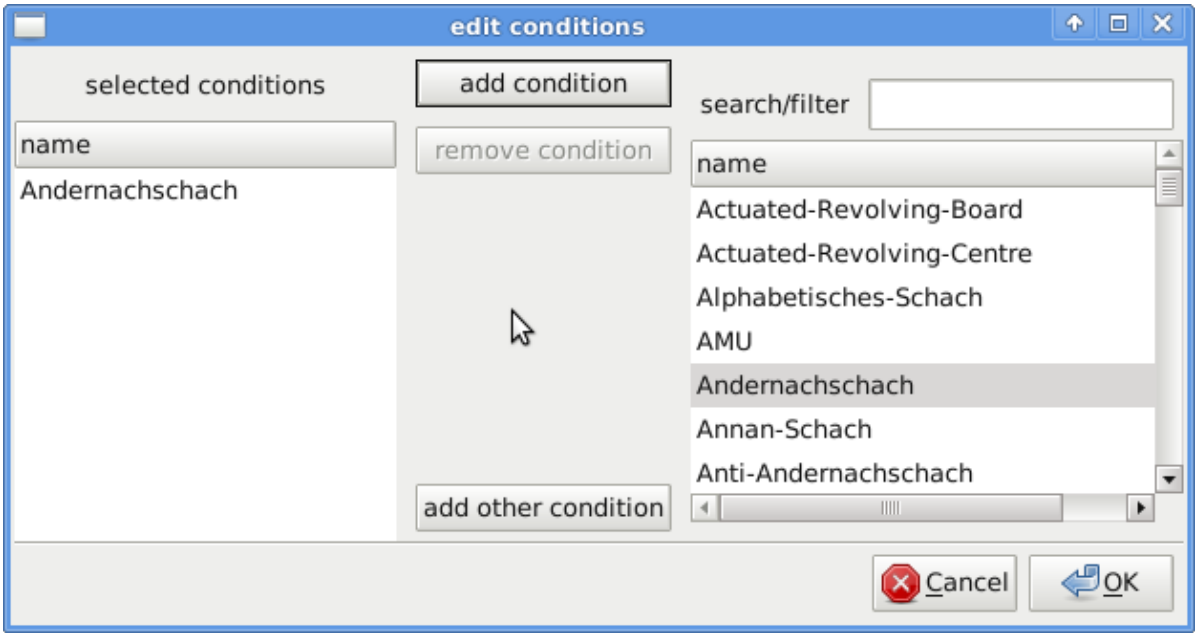

Abbildung 4: Die Auswahl von Bedingungen

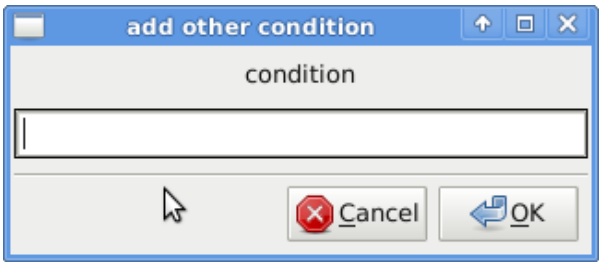

Abbildung 5: Die Auswahl von Bedingungen

Abbildung 6: Festlegung von Märchenfiguren

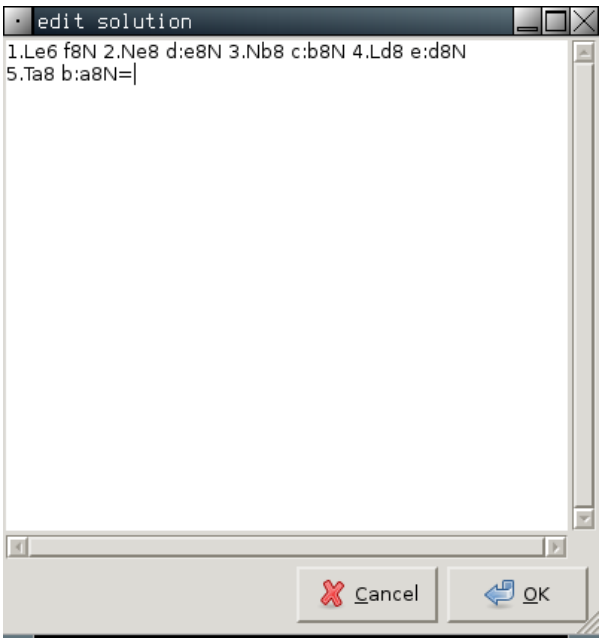

Abbildung 7: Editieren von Lösungen

Die Schaltflächen `gridchess` `verticalcylinder` and `horizontalcylinder`, `allwhite` und `switchcolors` dienen dazu, die entsprechenden visuellen elemente ein- oder auszuschalten.

Über die am unteren linken Rand dargestellten Felder können Figuren wieder entfernt werden (*clear*), Felder eingerahmt (*fieldframe*) sowie Felder entfernt werden (nofield) — letzteres macht hat natürlich nur sichtbare Auswirkungen bei schwarzen Feldern.

2.3 Das `Compile` Menu

Im Compile Menu wird ein LaTeX-Compile Vorgang angeboten, der das aktuelle Dokument in ein Benutzerdefiniertes Template einfügt und das Ergebnis des Compile Vorgangs anzeigt.

Die Konfiguration der Templates erfolgt in der Datei `config.py` im Konfigurationsverzeichnis des *ChessProblemEditor*.

Hierzu sind die folgenden Konfigurationen vorzunehmen:

Das Attribut

`DEFAULT_CONFIG.latex_compiler` muss den Namen des Programms enthalten, das zum Übersetzen von LaTeX Dateien benutzt wird.

Das Attribut

`DEFAULT_CONFIG.compiled_latex_viever` muss den Namen des Programms enthalten, das die vom LaTeX-Compiler erzeugten Dateien anzeigt.

Das Attribut

`DEFAULT_CONFIG.compile_menu_actions` mit einer Liste von Tupeln zu befüllen, die folgenden Inhalt haben:

1. Bezeichner im Menu
2. das Arbeitsverzeichnis für die Compile Vorgänge
3. der Name der Template Datei
4. der Name der Include-Datei — in diese Datei werden die Probleme des aktuellen Dokuments geschrieben.
5. der Name der Ausgabe Datei, die dann angezeigt wird.

Das Folgende zeigt eine solche Konfiguration:

```
_dias34_dir = join(expanduser('~'), '.cpe', 'dias34')

DEFAULT_CONFIG.compile_menu_actions = [
        ('fs dias3', _dias34_dir, 'dias3.tex', 'dias3.inc', 'dias3.dvi'),
        ('fs dias4', _dias34_dir, 'dias4.tex', 'dias4.inc', 'dias4.dvi')]
```

Die Variable `_dias34_dir` legt enthält das Arbeitsverzeichnis. Das erste Tupel besitzt einen Menu-Eintrag von `fs dias3`, der Name der Template Datei lautet `dias3.tex`, der Name der Include-Datei `dias3.inc` und die Ausgabedatei `dias3.dvi`. Alle diese Daten liegen im Arbeitsverzeichnis, bzw. werden dort erzeugt.

Die Datei `dias3.tex` hat dabei folgenden inhalt:

```
\documentclass{fs}

\newcommand{\rb}{}

\begin{document}
\begin{dialines}{3}{10pt}
\diagramxii

\input{dias3.inc}

\end{dialines}
\end{document}
```

2.4 Das Popeye Menu

Über das Popeye Menu können die aktuelle Aufgabe oder alle Aufgaben des Dokuments in Popeye Eingabe Syntax umgewandelt werden um dann mit Popeye gelöst zu werden.

Dieser Menu-Eintrag ist nur dann vorhanden, wenn in der Konfigurationsdatei der `DEFAULT_CONFIG.popeye_executable` Eintrag gemacht wurde.

Unter Windows sollte hier einfach das `py.exe` eingetragen werden.

Unter Unix kann hier ein Script eingetragen werden — bei mir verweist der Eintrag auf das folgende Script:

```
#!/bin/bash

LINES=40

nohup ${HOME}/bin/py $1 > $2 &
urxvt -title $2 -geometry 80x$LINES -e tail --lines=$LINES -f $2
```

Neben der Konfiguration der Executables kann noch das Verzeichnis eingestellt werden, das als *Arbeitsverzeichnis* für Popeye dient. In dieses Verzeichnis werden die Eingabe- und Ausgabe-Dateien geschrieben.

Ein Warnhinweis

Die Forderung wird - bis auf das entfernen des `\` vor einem `#` - derzeit einfach übernommen. Daher sollten die erzeugten Forderungen noch einmal kontrolliert und ggfs. angepasst werden.

Dazu ein Beispiel: wenn in der LaTeX-Datei `\stip{\#2*vv}` steht wird für Popeye die Zeile `forderung  #2*vv` erzeugt. Das `*vv` muss dann noch manuell entfernt bzw. durch `Option Satzspiel Verfuehrung` ersetzt werden.

2.5 Das Tools Menü

Der Menüpunkt *Tools* hat die folgenden Einträge:

- Mit *Fairy FEN* kann für das aktuelle Problem die *Fairy FEN* Notation erzeugt werden.
- Mit *Import Fairy FEN* kann Fairy FEN Notation erfasst (oder in den Dialog hineinkopiert) werden, die dann in die Stellung importiert wird. Ist die Stellung zu diesem Zeitpunkt nicht leer, so erfolgt eine Abfrage, ob die aktuelle Stellung überschrieben werden soll.
  Der Import der folgenden Fairy FEN Elemente ist derzeit **nicht** möglich:
  - C (Circle)
  - X (Cross)
  - S (Square)
  - T (Triangle)
  - Buchstaben und Zahlen auf den Feldern
- Mit *PNG Image* wird nach einem Dateinamen gefragt, unter dem dann eine PNG Bild-Datei mit der Stellung des aktuellen Problems erzeugt wird.

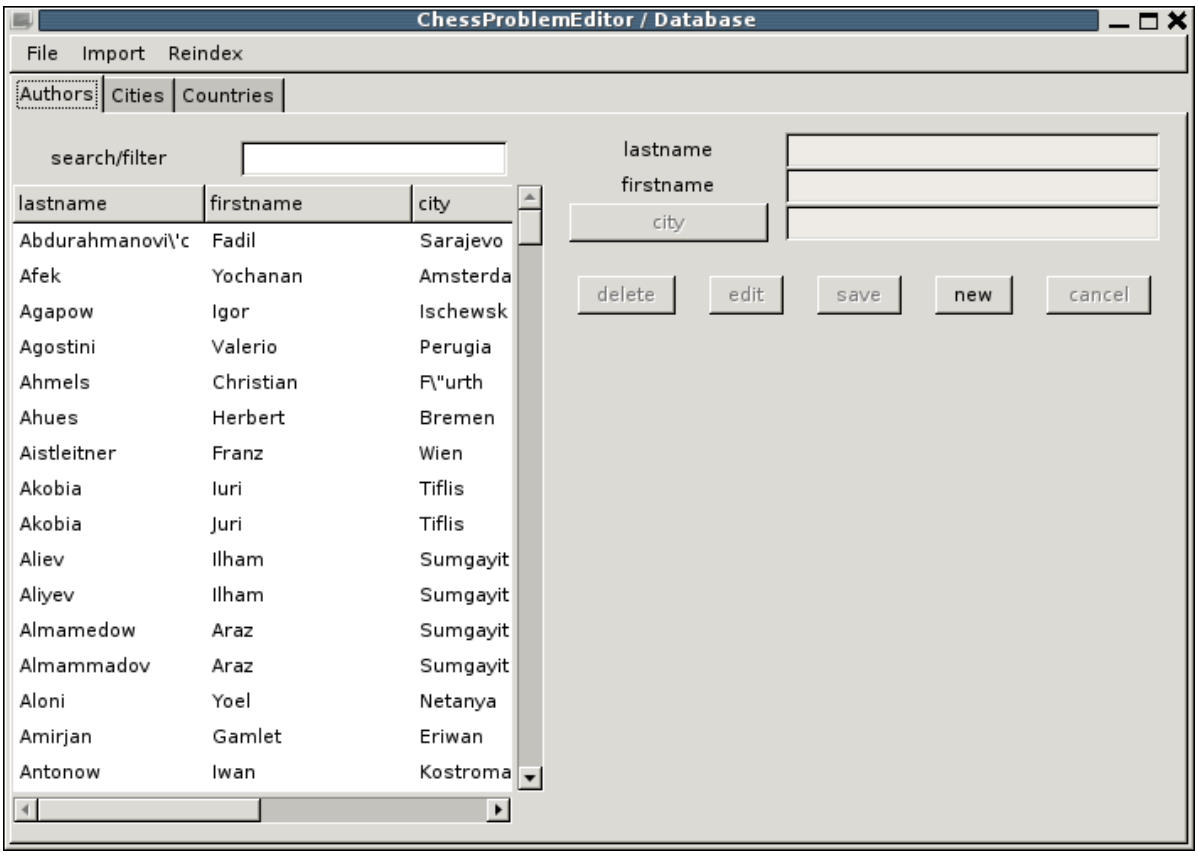

Abbildung 8: Das Hauptfenster der *ChessProblemDataBase*-Anwendung

3 Die Pflege der Autorendatenbank

Für die Pflege der Autorendaten steht eine separate Anwendung zur Verfügung, die mittels `cpdb.py` gestartet wird. Im Hauptfenster der Anwendung, das in Abbildung 8 dargestellt wird, gibt es Reiter für **Autoren**, **Städte** und **Länder**. Aussehen und Bedienung innerhalb der jeweiligen Reiter ist im wesentlichen gleich und soll daher nur für den Autorenreiter erläutert werden.

Über das `search/filter` Textfeld kann die Liste der angezeigten Autoren eingeschränkt werden. Dabei werden alle Autoren in der Liste angezeigt, die im Vor- oder Nachnamen den eingegebenen Text enthalten.[1][2]

Selektiert man einen Autor in dieser Liste, so erscheint er im Formular im rechten Bereich des Fensters. Zum Bearbeiten muss man hier den Button `edit` betätigen. Die Bearbeitung wird dann mit `save` abgeschlossen oder mit `cancel` wieder verworfen.

Ebenfalls ist hier die Neuanlage mit `new` oder das Löschen mit `delete` möglich.

[1] Im Detail ist das noch etwas komplizierter.

[2] Bei den Städten kommt ausserdem noch ein anderer Mechanismus — die Kölner Phonetik — zum Einsatz.

Option-Kurzform	Option-Langform	Bedeutung
-i	–issue	die Nummer des Hefts
-m	–month	der Erscheinungsmonat
-y	–year	das Erscheinungsjahr
-s	–start_sourcenr	die Nummer des ersten Urdrucks

Tabelle 2: Kommandozeilen Optionen beim `schwalbe_urdrucke.py`-Script

4 Utilities für die Bearbeitung der LaTeX-Dateien

Bestandteil der Software sind verschiedene Kommandozeilen Scripten, die bestimmte wiederkehrende Aufgaben automatisieren bzw. unterstützen. Diese sind im Folgenden beschrieben.

4.1 `schwalbe_urdrucke.py`

Das Script `schwalbe_urdrucke.py` dient dazu, nach Erscheinen eines Schwalbe-Heftes, die Quellenangaben sowie die Urdrucknummern bei den Aufgaben aus dem Urdruckteil zu ergänzen. Dazu müssen dem Script beim Aufruf die entsprechenden Werte als Kommandozeilenoptionen übergeben werden. Die notwendigen Optionen beschreibt Tabelle 2. Als weiteren Parameter muss man den Namen der Datei angeben. Es wird dann eine neue Datei mit der Extension `.new` geschrieben, die um die angebenen Informationen erweitert wurde.

4.2 `import_check.py`

Das Script `import_check.py` dient dazu, Autoren, Wohnorte und Länder, die in einer LaTeX-Datei vorkommen, vor dem Import zu kontrollieren.

Dabei wird für Länder kontrolliert, ob deren Kürzel bereits in der Datenbank enthalten sind und ob ggfs. nicht das **KFZ-Kennzeichen** sondern z. B. der **ISO-3166-3** Code benutzt wurde.

Bei den Städten wird kontrolliert, ob eine Stadt mit dem gleichen Namen in der Datenbank gespeichert ist. Eine Warnung wird ausgegeben, wenn eine solche Stadt gefunden wird, die aber — z. B. wg. anderer Akzente — in der Datenbank unterschiedlich geschrieben wurde, oder die Stadt gar nicht gefunden wurde.

Analog wird bei den Autoren verfahren.

4.3 `import_countries.py`

Dient dazu, die Tabelle der Länder initial aus einer CSV[3] Datei zu befüllen. Die CSV-Datei, die ich für die Befüllung benutzt habe, habe ich im wesentlichen aus dem Inhalt der

[3]CSV steht für *comma separated value*.

Website `http://www.aufenthaltstitel.de/staaten/schluessel.html` entnommen. Die aus diesen Daten erstellte CSV-Datei findet sich im Ordner `data`.

4.4 `import_sources.py`

Dieses Script dient dazu, die Namen von Quellen in die Datenbank zu importieren. Für jede Quelle soll die Datei eine Zeile mit dem Namen der Quelle enthalten. Eine Liste von Quellen, die aus entsprechenden Angaben in der *Schwalbe* erstellt wurde, befindet sich ebenfalls im Ordner `data` in der Datei `sources.csv`.

4.5 `cpd2ffen.py`

Dieses Script erzeugt für alle Probleme in der angegebenen Datei die entsprechende *Fairy FEN* Ausgabe der Stellung.

4.6 `cpd2popeye.py`

Dieses Script erzeugt für alle Probleme in der angegebenen Datei die Eingabe, die von Popeye zum Lösen benötigt wird.

4.7 `cpd2png.py`

Dieses Script erzeugt für alle Probleme in der angegebenen Datei eine PNG Bild-Datei mit dem Stellung.

5 Konfiguration der Software

Das Konfigurationsverzeichnis befindet sich innerhalb des HOME-Verzeichnisses des Benutzers im Unterverzeichnis `.cpe`.[4]

Sollte dieses Verzeichnis, sowie die benötigten Dateien noch nicht vorhanden sein, so werden alle beim Programmstart automatisch angelegt. Details zur Konfiguration sind (auf englisch) in der Datei `config.py` beschrieben.

Ebenfalls dort erzeugt wird dann eine Datei `condition_config.py` die eine Liste der dem Programm bekannten Namen für Märchenbedingungen enthält. Diese Liste enthält Einträge der Form

```
Condition('<name>', '<popeye-schluesselwort>')
```

Dabei ist `name` der in der Anwendung und in den LaTeX-Dateien benutze Text und `popeye schluesselwort` die Bedingung, die beim Aufruf von Popeye zu verwenden ist.

Weiterhin wird dort ein Datei `piecetype_config.py` angelegt, die eine Liste der Märchenfiguren enthält. Die Liste enthält Einträge der Form:

```
PieceDef('<name>', '<popeye-figuren-kuerzel>')
```

[4] Unter Windows wird dies durch die Umgebungsvariablen `HOMEDRIVE` und `HOMEPATH` festgelegt.

6 Installation

6.1 Vorbereitung

Um die Software installieren zu können muss man auf seinem Rechner einen *Python* Interpreter installiert haben, sowie die notwendigen Bibliotheken um *GTK 3* Programme laufen zu lassen.

6.1.1 Linux

Bei den meisten Linux Distributionen sind Python und die GTK 3 Bibliotheken (`python-gobject`) als Pakete vorhanden, die mit dem Paketmanager der Distribution installiert werden können.

6.1.2 Windows

Die Installationsdateien zu *Python* findet man auf `http://www.python.org/download/`. Die Software benötigt Python 3. Beim Installieren sollte man darauf achten, dass die Verzeichnisse mit den EXE Dateien mit in den Pfad aufgenommen werden.

Die Benutzeroberfläche ist in GTK 3 erstellt. Unter ist der aktuelle Installer bei sourceforge erhältlich:
`http://sourceforge.net/projects/pygobjectwin32/files/pygi-aio-3.14.0_rev12-setup.exe/`

Bei der Installation ist darauf zu achten, dass im Installationsdialog Choose available GNOME/Freedesktop libraries to install.ßusätzlich zum vorausgewählten Eintrag die beiden Einträge "GDK-Pixbuf 2.30.8ünd "GTK+ 3.14.8äusgewählt werden müssen.

6.2 Download und Installation der ChessProblemEditor-Software

Die Software kann mit Python mitteln, oder manuell installiert werden. Zur Installation mit Python mitteln wird das Programm `easy_install` genutzt, das dabei auf Informationen aus dem *Python Package Index* `http://pypi.python.org/` zugreift. Zur Installation muss man dann folgendes eingeben:

```
easy_install chessproblem.ui
```

Statt des Paketnamens kann man auch direkt die URL angeben, von dem das Source-Paket heruntergeladen wird:

```
easy_install http://garr.dl.sourceforge.net/project/chessproblemed/chessproblem.ui-0.
```

Dieses Paket kann man auch benutzen, um die Software manuell zu installieren. Dies geschieht wiefolgt: man muss die Datei in einem Verzeichnis seiner Wahl entpacken und dann anschliessend in das Unterverzeichnis `chessproblem.ui` dieses Verzeichnisses wechseln.

Hier muss man dann folgendes eingeben:

```
python setup.py install
```

Danach kann man die *ChessProblemEditor*-Programme starten.

7 Die Organisation und Pflege des Sourcecodes

Die gesamten Sourcen — inclusive des LaTeXSourcecodes dieser Dokumentation, sind auf *Sourceforge* verfügbar. Der Link lautet:
`https://sourceforge.net/projects/chessproblemed/`

Zur Organisation von Änderungen in der Software benutze ich **hg**. Damit ist nicht Hans Gruber sondern das verteilte Versionskontrollsystem *Mercurial*[5] gemeint.

Für die Verwaltung von Problemen benutze ich **be**. Hier ist nicht bernd ellinghoven sondern das Issue-Tracking-System *bugs everywhere*[6] gemeint.

Die Entscheidung für diese beiden Werkzeuge war aber sicherlich davon beeinflusst, dass die Befehle mit den Kürzeln von bernd und Hans übereinstimmen.

8 Copyright / Lizenzen

8.1 Copyright und Lizenz dieser Software

Diese Software ist unter der *GNU General Public License* veröffentlicht, deren Wortlaut in der Datei `gpl.txt` enthalten ist.

Im folgenden finden sie den Copyright vermerk, der in daher jeder Datei[7] der *ChessProblemEditor*-Software enthalten ist:

```
Copyright 2012 Stefan Hoening

This file is part of the "Chess-Problem-Editor" software.

Chess-Problem-Editor is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

Chess-Problem-Editor is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see
http://www.gnu.org/licenses/

Diese Datei ist Teil der Software "Chess-Problem-Editor"
```

[5] `http://mercurial.selenic.com/`
[6] `http://bugseverywhere.org/`
[7] Ausgenommen sind natürlich Binärdateien wie z. B. die Figuren-Images.

```
Chess-Problem-Editor ist Freie Software: Sie koennen es unter den Bedingungen
der GNU General Public License, wie von der Free Software Foundation,
Version 3 der Lizenz oder (nach Ihrer Option) jeder spaeteren
veroeffentlichten Version, weiterverbreiten und/oder modifizieren.

Chess-Problem-Editor wird in der Hoffnung, dass es nuetzlich sein wird, aber
OHNE JEDE GEWAEHRLEISTUNG, bereitgestellt; sogar ohne die implizite
Gewaehrleistung der MARKTFAEHIGKEIT oder EIGNUNG FUER EINEN BESTIMMTEN ZWECK.
Siehe die GNU General Public License fuer weitere Details.

Sie sollten eine Kopie der GNU General Public License zusammen mit diesem
Programm erhalten haben. Wenn nicht, siehe
http://www.gnu.org/licenses/
```

8.2 Lizenzen benutzter Bibliotheken

Innerhalb der *ChessProblemEditor*-Software werden weitere Bibliotheken eingesetzt, die ebenfalls OpenSource sind. Tabelle 3 enthält die Namen und Versionen der benutzten Bibliotheken sowie deren Lizenz.

Name	**Version**	**Lizenz**
setuptools	0.6.14	PSF or ZPF
sqlalchemy	0.7.3	MIT
Pygments	1.4	BSD
kph	0.1	GPL
python-Levenshtein	0.10.2	GPL

Tabelle 3: Benutzte Open Source Bibliotheken