

Teil I

Packages und Klassen

In den folgenden Abschnitten sollen die Packages und die darin enthaltenen Klassen und Methoden näher erläutert werden.

1 Package de.BibTexMaker

Das Package *de.BibTexMaker* beinhaltet das gesamte Programm BIBTEXMAKER. Die darin enthaltenen Packages und Klassen werden im folgenden aufgezählt und deren Funktion erläutert.

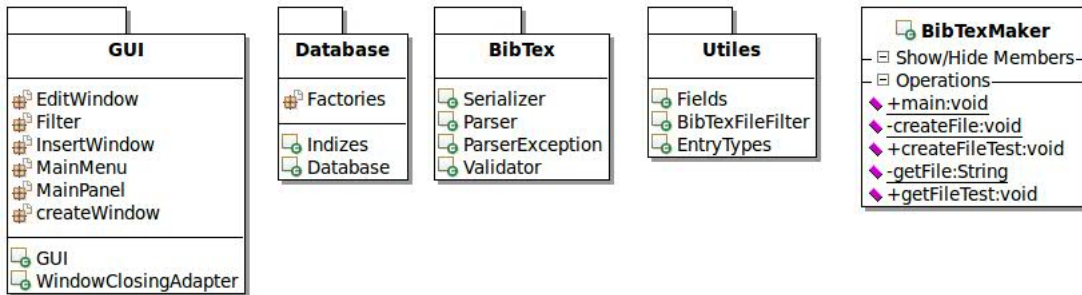


Abbildung 1: Klassendiagramm Package *de.BibTexMaker*

1.1 Klasse BibTexMaker

Die Klasse *BibTexMaker* beinhaltet die Methoden *main*, *createFile* und *getFile*. Die Klasse bildet den Einstieg in das Programm BIBTEXMAKER

Die Methode *main* ist der Einstieg in das Programm BIBTEXMAKER. Sie prüft ob Argumente übergeben wurden und reagiert entsprechend. Wurde ein Dateiname mit Pfad für eine BibTexMaker-Datenbank-Datei übergeben, so veranlasst die Methode das Laden dieser. Ansonsten wird eine leere Datenbank (`./library.xml`) angelegt. Dabei greift die Methode *main* auf die Methode *getFile* zu und übergibt dieser die Argumentenliste.

Die Methode *getFile* liest die dem Programm übergebenen Parameter aus und wertet den Parameter *-f* aus und gibt den Dateinamen mit Pfadangabe zurück. Der Parameter *-f* ist dafür gedacht eine bereits vorhandenen Datenbankdatei direkt beim Programmstart zu laden. Wurde kein Parameter *-f* übergeben, so wird der Methode *main* ein leerer String zurückgegeben.

Die Methode *main* prüft nun ob die Datei existiert. Existiert die Datei noch nicht wird diese erzeugt. Dazu wird die Methode *createFile* aufgerufen.

Die Methode *createFile* erstellt eine neue Datenbankdatei mit dem entsprechenden Format. Anschließend übergibt die Methode *main* den Pfad und den Dateinamen der Datenbank, so dass diese die Daten laden kann.

Zum Schluss startet die Methode *main* die GUI.

1.2 Package de.BibTexMaker.GUI

Das Package *de.BibTexMaker.GUI* beinhaltet alle Packages und Klassen welche zum Graphical User Interface (GUI) gehören.

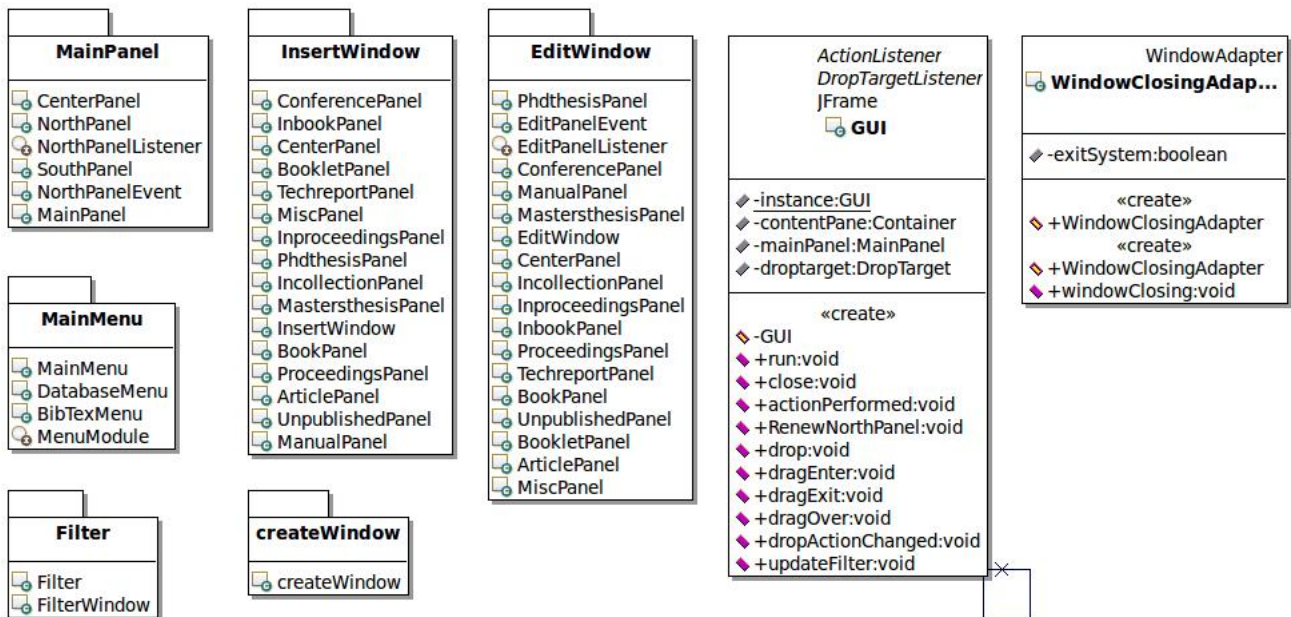


Abbildung 2: Klassendiagramm Package *de.BibTexMaker.GUI*

1.2.1 Klasse GUI

Die Klasse *GUI* ist, wie die Abkürzung schon vermuten lässt, das Graphic Unit Interface des Programms bzw. beinhaltet alle notwendigen Methoden, um dieses darzustellen. Die Klasse ist nach dem Singleton-Pattern-Verfahren programmiert. Das heißt es kann im Programm nur eine Instanz der Klasse *GUI* erzeugt werden. Dazu wurde der Konstruktor *private* gekennzeichnet und eine Methode *getInstance* implementiert. Mithilfe der

Methode *getInstance* kann auf die Instanz der Klasse zugegriffen werden. Ist die Instanz noch nicht angelegt, wird Sie durch die Methode *getInstance* erzeugt.

Mithilfe der Methode *run* wird die GUI zur Anzeige gebracht. Die Methode stellt dabei die Größe des Bildschirms fest und berechnet entsprechend die Größe des Hauptfensters und setzt dieses in die Mitte des Bildschirms.

Die Methode *close* kann dazu genutzt werden einen `Window_Closing-Event` auszulösen, um damit das Fenster oder das Programm zu beenden.

Mit der Methode *RenewNorthPanel* wird veranlasst, dass das `NorthPanel` neu aufgebaut wird und somit aktualisiert wird. Das `NorthPanel` beinhaltet die Buttons, die Comboboxen zur Auswahl des Publikationstypen usw.

Die Methode *drop* reagiert auf einen `Drop-Event` und veranlasst das Prüfen des übergebenen Elementes. Liegt hier eine BibTex-Datei vor, so wird diese an den Parser weitergeleitet, damit die enthaltenden Daten in die Datenbank aufgenommen werden.

Die Methode *updateFilter* löst das aktualisieren des Filters aus.

Alle anderen Methoden haben keine Funktion, sondern sind vorhanden, da dies die Implementation der Interfaces `ActionListener` und `DropTargetListener` erfordert.

1.2.2 Klasse `WindowClosingAdapter`

Die Klasse `WindowClosingAdapter` dient zur Entscheidung, ob nur das Hauptfenster geschlossen werden soll oder das gesamte Programm beendet werden soll, wenn ein `Window_Closing-Event` ausgelöst wird.

Dazu kann die Klasse über zwei Konstruktoren instantiiert werden. Wird ein Objekt der Klasse *WindowClosingAdapter* ohne Übergabe des Parameters *exitSystem* übergeben, so wird bei Auslösen des `Window_Closing-Events` lediglich das Hauptfenster geschlossen. Wird der Parameter übergeben, dann wird bei `true` auch das Programm beendet und nicht nur das Hauptfenster geschlossen.

Wird nun ein `Window_Closing-Event` ausgelöst, dann wird die Methode *windowClosing* ausgeführt. Dabei schließt Sie das Filterfenster falls geöffnet und das Hauptfenster. Anschließend wird geprüft ob das Attribut *exitSystem* auf `true` gesetzt wurde. Ist dies der Fall veranlasst die Methode das Beenden des Programmes.

1.2.3 Package MainPanel

Das Package *MainPanel* beinhaltet alle Klassen zum Aufbau des Hauptfensters.

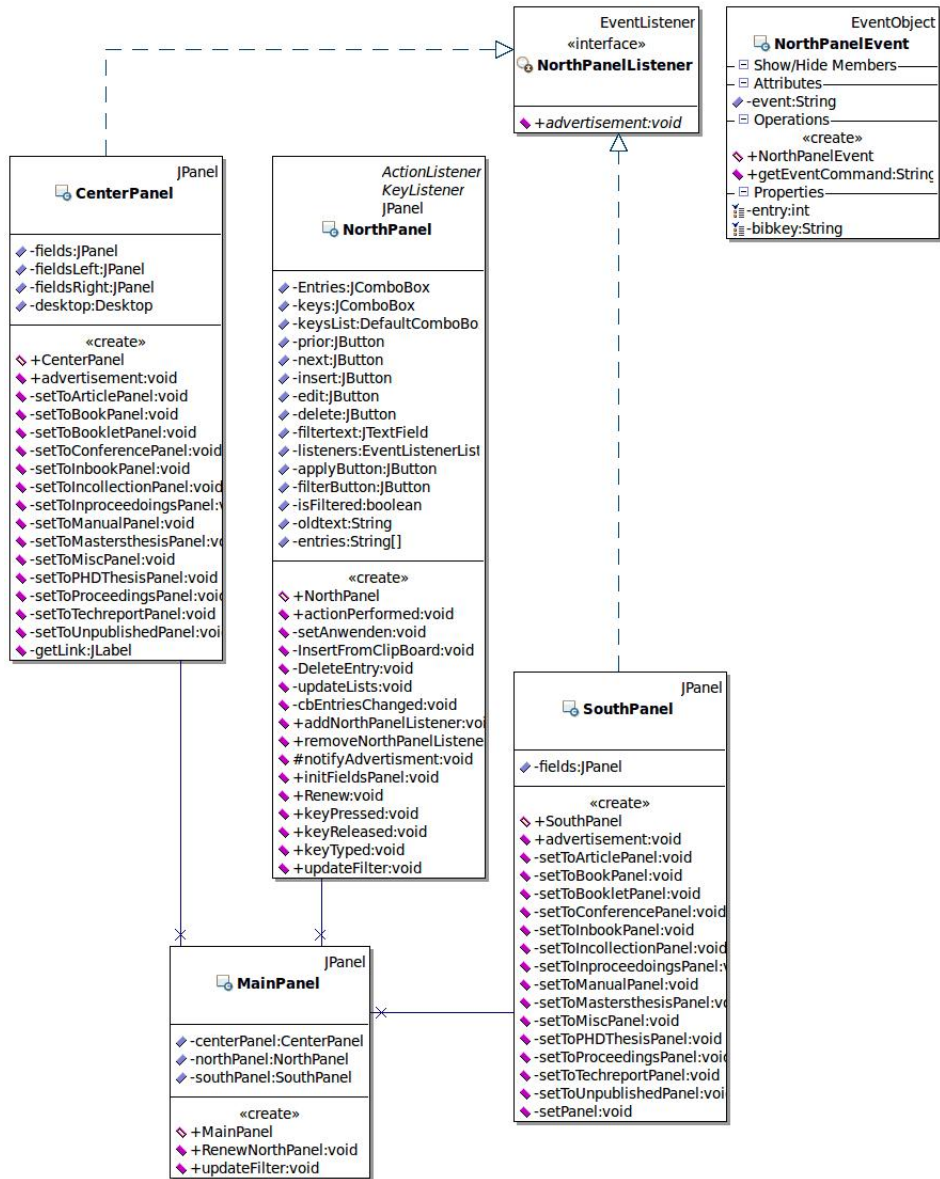


Abbildung 3: Klassendiagramm Package *de.BibTexMaker.GUI.MainPanel*

1.2.3.1 Klasse MainPanel

Die Klasse MainPanel beinhaltet alle Methoden zum Aufbau des Hauptfensters. Dazu wird im Konstruktor das Hauptfenster initialisiert. Dazu wird das Layout und die Border festgelegt. Als Layout wird das BorderLayout verwendet. An die Position North kommt das NorthPanel, Center das CenterPanel und an die Position South das SouthPanel. Es werden als Attribut jeweils ein Objekt des North-, Center- und SouthPanel angelegt.

Des Weiteren meldet der Konstruktor das CenterPanel und das SouthPanel am NorthPanelListener an, damit diese auf NorthPanelEvents reagieren können.

Zum Schluss wird die Methode *initFieldsPanel* des NorthPanels ausgeführt, um die Comboboxen für den Publikationstyp und der Auswähl der Publikation zu füllen.

Zur Klasse *NorthPanel* gehören auch noch die Methoden *RenewNorthPanel* und *updateFilter*, welche das Aktualisieren des Filterfensters und des NorthPanels veranlassen.

Das MainPanel ist eine abgeleitete Klasse von JPanel.

1.2.3.2 Klasse NorthPanel

Die Klasse NorthPanel erzeugt das NorthPanel und kümmert sich um die Ausführung der entsprechenden Methode bei betätigen eines Buttons oder der Änderung eines Eintrages in einer der Comboboxen. Im Konstruktor werden die einzelnen Elemente der Klasse initialisiert. Dabei wird innerhalb der Klasse das Panel in zwei weitere Panels geteilt. Das obere und das untere Panel. Das obere Panel beinhaltet die Auswahlliste für Publikationstyp und Publikation, sowie die Schaltflächen VORHERIGER, NÄCHSTER, HINZUFÜGEN, BEARBEITEN und LÖSCHEN. Das untere Panel beinhaltet die Elemente des Filters. Dazu gehören das Editierfeld für den Filtertext und die Schaltflächen FELDER und ANWENDEN / ABSCHALTEN.

Wird in der Auswahlliste der Publikationstypen ein anderer Publikationstyp ausgewählt, so wird die entsprechende Methode *actionPerformed* ausgeführt. Die Methode führt die Methode *cbEntriesChanged* aus. Anschließend wird die Auswahl in der Combobox neu festgelegt. und ein NorthPanelEvent ausgelöst, welcher veranlasst die Änderung in der Combobox zu übernehmen.

Wählt der Anwender in der Auswahlliste der Publikationen eine andere Publikation aus, so wird auch hier die zugehörige Methode *actionPerformed* aufgerufen. Die Methode löst den NorthPanelEvent aus und teilt so den darauf hörenden Methoden mit, dass sich der Datensatz in der Auswahl geändert hat.

Betätigt der Anwender eine der Schaltflächen, wird die entsprechende Methode *actionPerformed* ausgeführt. Diese Methode prüft nun, welche Schaltfläche betätigt wurde.

Wurde die Schaltfläche HINZUFÜGEN betätigt, so erfolgt zunächst eine Abfrage ob Datensätze aus der Zwischenablage in die Datenbank aufgenommen werden sollen oder ob von Hand ein neuer Datensatz erstellt werden soll. Dies geschieht über ein JOptionPane-Objekt. Anschließend wird die Methode *InsertFromClipboard* ausgeführt, wenn der Anwender sich entschieden hat von der Zwischenablage die Daten einzulesen, ansonsten wird ein Objekt vom Typ *InsertWindow* erstellt und mit der Methode *run* geöffnet. Anschließend wird die Methode

updateLists ausgeführt, um die Änderungen in der Datenbank in der GUI anzuzeigen, also zu übernehmen.

Hat der Anwender die Schaltfläche BEARBEITEN betätigt, wird zunächst geprüft, ob ein Eintrag ausgewählt bzw. überhaupt vorhanden ist, welcher bearbeitet werden kann. Ist dies der Fall so wird eine Instanz des Fensters zum Bearbeiten von Datensätzen (EditWindow) angelegt. Dabei wird dem Konstruktor der BibKey und der Publikationstyp des zu bearbeitenden Datensatzes übergeben. Anschließend wird das Fenster geöffnet. Nachdem der Anwender das Fenster geschlossen hat, wird noch das Aktualisieren der entsprechenden Panels veranlasst.

Wenn die Schaltfläche LÖSCHEN betätigt wurde, wird die Methode *DeleteEntry* aufgerufen, welche die ausgewählte Publikation aus der Datenbank entfernt.

Bei betätigen der Schaltfläche VORHERIGER, wird geprüft ob der selektierte Eintrag in der Combobox nicht der Erste ist. Ist dies der Fall so wird die Selektierung auf den davor liegenden Eintrag gesetzt. Die Combobox löst somit einen Event aus der die darauf hörende Methode zu entsprechenden Änderungen veranlasst.

Bei betätigen der Schaltfläche NÄCHSTER, wird geprüft ob der selektierte Eintrag in der Combobox nicht der Letzte ist. Ist dies der Fall so wird die Selektierung auf den danach liegenden Eintrag gesetzt. Die Combobox löst somit einen Event aus der die darauf hörende Methode zu entsprechenden Änderungen veranlasst.

Wird die Schaltfläche FELDER betätigt, so wird das Filterfenster (FilterWindow) aufgerufen.

Betätigt der Anwender die Schaltfläche ANWENDEN/ABSCHALTEN wird mithilfe der Methode *setAnwenden* der Filter ein- oder ausgeschaltet.

Die Methode *setAnwenden* prüft zunächst ob der Filter bereits zugeschaltet ist, wenn dem der Fall ist, wird er abgeschaltet, ansonsten eingeschaltet. Beim Einschalten wird der Status in *isFiltered* auf true gesetzt, der Filter-Button erhält die Beschriftung *Abschalten* und den ToolTipText *Filter abschalten*. Wird jedoch der Filter abgeschaltet, so wird der Status in *isFiltered* auf false gesetzt und der Filter-Button erhält die Beschriftung *Anwenden* und der ToolTipText *Filter anwenden*. Weiterhin wird das Attribut *oldtext* gelöscht und dem Editierfeld des Filters zugewiesen. Sowohl beim Ein- bzw. Ausschalten, wird zum Schluss die Methode *cbEntriesChanged* ausgeführt, welche die ComboBox mit den Publikationseinträgen neu aufbaut.

Die Methode *InsertFromClipboard* veranlasst alle Maßnahmen, um aus der Zwischenablage Datensätze in die Datenbank zu übernehmen. Dabei werden zunächst die Daten aus der Zwischenablage abgeholt und an den Parser weitergeleitet. Dieser wandelt die Datensätze in Datenbankdaten um und führt die Validierung durch, wo die Daten auf Korrektheit geprüft werden. Zum Schluss wird noch die Methode *Renew* aufgerufen, welche veranlasst, dass das NorthPanel neu aufgebaut wird, um damit eine Aktualisierung des Panels zu erreichen.

Die Methode *DeleteEntry* kümmert sich darum das ein Datensatz aus der Datenbank entfernt wird. Zunächst erfolgt eine Sicherheitsabfrage, ob der ausgewählte Datensatz auch wirklich gelöscht werden soll. Bestätigt der Anwender dies, wird das Löschen veranlasst. Dazu wird der BibKey des zu löschenden Datensatzes ermittelt und den Methoden *delFromAll* und *delFile* aus der Klasse *Indizes* übergeben. Diese Methoden sorgen dafür, dass aus den entsprechenden Listen, welche für die Combobox *keys* verwendet wird entfernt werden. Anschließend wird die Datenbank veranlasst den Datensatz aus der Datenbank zu entfernen. Wurde der Datensatz aus der Datenbank entfernt, wird die Datenbank über den Marshaller dazu aufgefordert, die Datensätze in der entsprechenden Datenbankdatei zu überarbeiten. Das heißt die Datenbank auf der Festplatte wird gelöscht und

die Datensätze neu aufgebaut. Anschließend wird die Methode *Renew* der Datenbank aufgerufen und mit der Methode *cbEntriesChanged* veranlasst, die Combobox *keys* aktualisiert wird.

Die Methode *updateList* prüft zunächst ob eine Publikation vorhanden ist bzw. ausgewählt ist. Ist dies der Fall wird der BibKey des ausgewählten Eintrages ermittelt und das aktualisieren der Combobox *keys* mithilfe der Methode *cbEntriesChanged* veranlasst. War ein BibKey zu ermitteln wird zum Schluss die Selektion der Combobox *keys* auf diesen Eintrag gesetzt.

Die Methode *cbEntriesChanged* dient zum Erneuern der Liste in der Combobox *keys*. Die Combobox *keys* enthält die aufrufbaren Publikationen je nach Auswahl des Publikationstypen in der Combobox *Entries*.

Die Methode *addNorthPanelListener* dient anderen Klassen bzw. Methoden dazu sich in den *NorthPanelListener* einzutragen, um bei entsprechenden Events informiert zu werden.

Die Methode *removeNorthPanelListener* dient anderen Klassen bzw. Methoden dazu sich aus dem *NorthPanelListener* auszutragen.

Soll ein *NorthPanelEvent* ausgelöst werden, so muss die Methode *notifyAdvertisement* aufgerufen werden. Diese geht dann die Liste der Klassen bzw. Methoden durch welche sich am *NorthPanelListener* angemeldet haben und informiert diese, dass ein *NorthPanelEvent* ausgelöst wurde.

Mithilfe der Methode *initFieldsPanel* kann der *NorthPanelEvent KeyChanged* ausgelöst werden. Dazu wird der ausgewählte Publikationstyp und die ausgewählte Publikation ermittelt und dem Event hinzugefügt.

Über die Methode *Renew* können externe Methoden das Aktualisieren der Liste mit den Publikationen (*keys*) veranlassen. Dazu wird die Methode *updateLists* aufgerufen.

Die Methode *keyPressed* reagiert auf einen Tastendruck im Editierfeld des Filters. Dabei wird geprüft ob ein Text eingegeben ist und die Taste BACKSPACE gedrückt wurde. Wurde die Taste BACKSPACE gedrückt und ist nur noch ein Buchstabe vorhanden, so wird der Filtertext gelöscht und die Methode *cbEntriesChanged* ausgeführt.

Die Methode *keyReleased* reagiert auf das loslassen einer Taste im Editierfeld des Filters. Dabei wird geprüft, ob sich der Filtertext geändert hat. Hat er sich geändert wird *cbEntriesChanged* aufgerufen, um die Auswahlliste der Publikationen zu aktualisieren. Bei eingeschaltetem Filter veranlasst die Methode die Filterung der Publikationen bevor diese in die Liste eingetragen werden.

Die Methode *updateFilter* kann von externen Methoden dazu verwendet werden um, bei eingeschaltetem Filter einen Neuaufbau der Auswahlliste der Publikationen zu veranlassen.

1.2.3.3 Interface *NorthPanelListener*

Das Interface *NorthPanelListener* wird von den Klassen eingebunden, welche auf *NorthPanelEvents* hören. Es ist von *EventListener* abgeleitet und führt die Methode *advertisement* ein. Diese Methode wird in den entsprechenden Klassen ausgeführt, wenn ein *NorthPanelEvent* ausgelöst wurde und die Klassen sich am Listener angemeldet haben.

1.2.3.4 Klasse NorthPanelEvent

Die Klasse *NorthPanelEvent*, wird dazu genutzt, um bei einem auslösen eines *NorthPanelEvents* ein Objekt dieser Klasse zu erzeugen. Die Klasse ist von *EventObject* abgeleitet und erweitert diese Klasse, um die Informationen des Publikationstyps und des Index der Publikation.

1.2.3.5 Klasse CenterPanel

Die Klasse *CenterPanel* erzeugt das CenterPanel und dient zur Anzeige der Informationen zu einer ausgewählten Publikation. Die Klasse ist von *JPanel* abgeleitet und implementiert den *NorthPanelListener*, um auf *NorthPanelEvents* reagieren zu können (z.B. Änderung der Publikationsauswahl).

Die Klasse teilt das CenterPanel nochmal in zwei Panel auf. Das linke Panel (*fieldsLeft*) und das rechte Panel (*fieldsRight*). Im linken Panel werden die Labels zu den Feldern eingefügt und im rechten Panel die dazugehörigen Informationen der ausgewählten Publikation. Welcher Felder angezeigt wird entscheidet sich über den Publikationstyp der ausgewählten Publikation.

Damit immer die aktuellen Daten zur jeweilig ausgewählten Publikation angezeigt werden, hört die Klasse auf *NorthPanelEvents*. Wird solch ein Event ausgelöst wird die Methode *advertisement* aufgerufen. Dabei prüft die Methode ob der Event mit Kommando *KeyChanged* ausgelöst wurde. Ist dies der Fall veranlasst die Methode das Löschen aller Elemente in den Panels *fieldsLeft*, *fieldsRight* und *fields*. Somit ist das CenterPanel erst einmal leer. Anschließend fragt die Methode beim Event den BibKey ab. Wurde keiner mit dem Event mitgeliefert, so bleibt das CenterPanel leer. Wurde ein BibKey mitgeliefert so wird der Publikationstyp der Publikation abgefragt. Entsprechend des Publikationstyps werden die Panels *fieldsRight* und *fieldsLeft* über eine weitere Methode initialisiert und mit den Daten der Publikation befüllt. Zum Schluss werden die Panels dem Panel *fields* übergeben und das Panel *fields* wird veranlasst sich neu aufzubauen.

1.2.3.6 Klasse SouthPanel

Die Klasse *SouthPanel* erzeugt das SouthPanel. Es werden Methoden bereitgestellt welche die Initialisierung der Elemente des SouthPanels sicherstellen und deren Aktualität in Hinsicht auf deren Informationen zur ausgewählten Publikation. Das SouthPanel ist von *JPanel* abgeleitet.

Die Klasse hört auf *NorthPanelEvents*. Wird ein solcher Event ausgelöst, wird die Methode *advertisement* aufgerufen. Wurde dabei ein *NorthPanelEvent* mit Kommando *KeyChanged* ausgelöst, veranlasst die Methode das Löschen aller Elemente im Panel *fields*. Somit ist das SouthPanel erst einmal leer. Anschließend fragt die Methode beim Event den BibKey ab. Wurde keiner mit dem Event mitgeliefert, so bleibt das SouthPanel leer. Wurde ein BibKey mitgeliefert so wird der Publikationstyp der Publikation abgefragt. Entsprechend des Publikationstyps wird das Panel *fields* über eine weitere Methode initialisiert und mit den BibTex-Dateinamen und Pfadangaben der Publikation befüllt. Zum Schluss wird das Panel angewiesen sich neu aufzubauen.

1.2.4 Package MainMenu

Das Package *de.BibTex.GUI.MainMenu* enthält alle Klasse die das Hauptmenü repräsentieren.

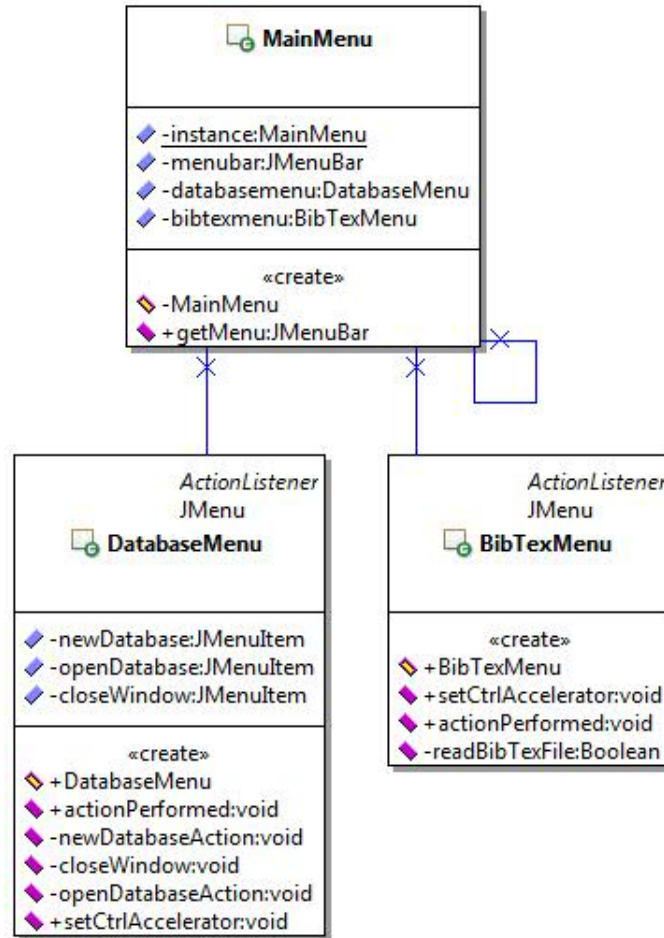


Abbildung 4: Klassendiagramm Package *de.BibTexMaker.GUI.MainMenu*

1.2.4.1 Klasse MainMenu

Die Klasse *MainMenu* stellt das Hauptmenü des Programmes zur Verfügung. Es bindet dazu Instanzen der Klassen *BibTexMenü* und *DatenbankMenü* ein.

Die Methode *getMenu* übergibt der aufrufenden Methode das Hauptmenü.

1.2.4.2 Klasse DatabaseMenu

Die Klasse *DatabaseMenu* stellt das Untermenü für die Datenbankfunktionen dem Hauptmenü zur Verfügung. Die Klasse ist von *JMenu* abgeleitet und implementiert den *ActionListener*.

Die Methode *actionPerformed* wird ausgeführt, wenn ein Menüeintrag ausgewählt wurde. Je nachdem welcher Menüeintrag ausgewählt wurde wird die jeweilige Methode (*newDatabaseAction*, *openDatabaseAction*, *closeWindow*) ausgeführt.

Die Methode *newDatabaseAction* legt eine neue Datenbankdatei an. Dazu öffnet sie ein Auswahlfenster bei der das Verzeichnis gewählt werden kann, in welches die Datei erzeugt werden soll. Des Weiteren wird über das Fenster der Dateiname angegeben. Wurde ein Dateiname übergeben, so erzeugt die Methode ein File-Objekt und übergibt dieses der Database-Methode *newDatabaseFile*. Diese Methode erstellt nun die Datenbankdatei.

Die Methode *openDatabaseAction* öffnet eine Datenbankdatei. Dazu öffnet sie ein Auswahlfenster mit dessen Hilfe auf einem Speichermedium eine Datenbankdatei ausgewählt werden kann. Wurde eine Datei ausgewählt, so wird diese der Datenbank übergeben, so dass die Datenbank die Daten laden kann.

Die Methode *closeWindow* veranlasst über die Methode *close* der Klasse *GUI*, dass das Programm beendet wird.

1.2.4.3 Klasse BibTexMenu

Die Klasse *BibTexMenu* stellt das Untermenü für die BibTex-Funktionen dem Hauptmenü zur Verfügung. Die Klasse ist von *JMenu* abgeleitet und implementiert den *ActionListener*.

Die Methode *actionPerformed* wird ausgeführt, wenn ein Menüeintrag ausgewählt wurde. Die Methode prüft welches Kommando übergeben wurde. Wurde das Kommando *create* übergeben, so wird eine Instanz der Klasse *createWindow* erzeugt und das Fenster geöffnet. Wurde jedoch das Kommando *read* übergeben, so wird die Methode *readBibTexFile* ausgeführt und anschließend das NorthPanel angewiesen sich neu aufzubauen, um sich dadurch zu aktualisieren.

Die Methode *readBibTexFile* öffnet ein Auswahlfenster mit der BibTex-Dateien auf einem Speichermedium ausgewählt werden können. Diese Auswahl wird dann an den Parser weitergeleitet, welcher die Daten der BibTex-Dateien in die Datenbank aufnimmt. Vorher wird noch geprüft, ob eine der ausgewählten BibTex-Dateien schon in der Datenbank enthalten ist. Ist dies der Fall wird der gesamte Vorgang mit einer entsprechenden Meldung abgebrochen.

1.2.5 Package Filter

Das Package *de.BibTexMaker.GUI.Filter* enthält zwei Klassen. Die Klasse *Filter* und die Klasse *FilterWindow*.

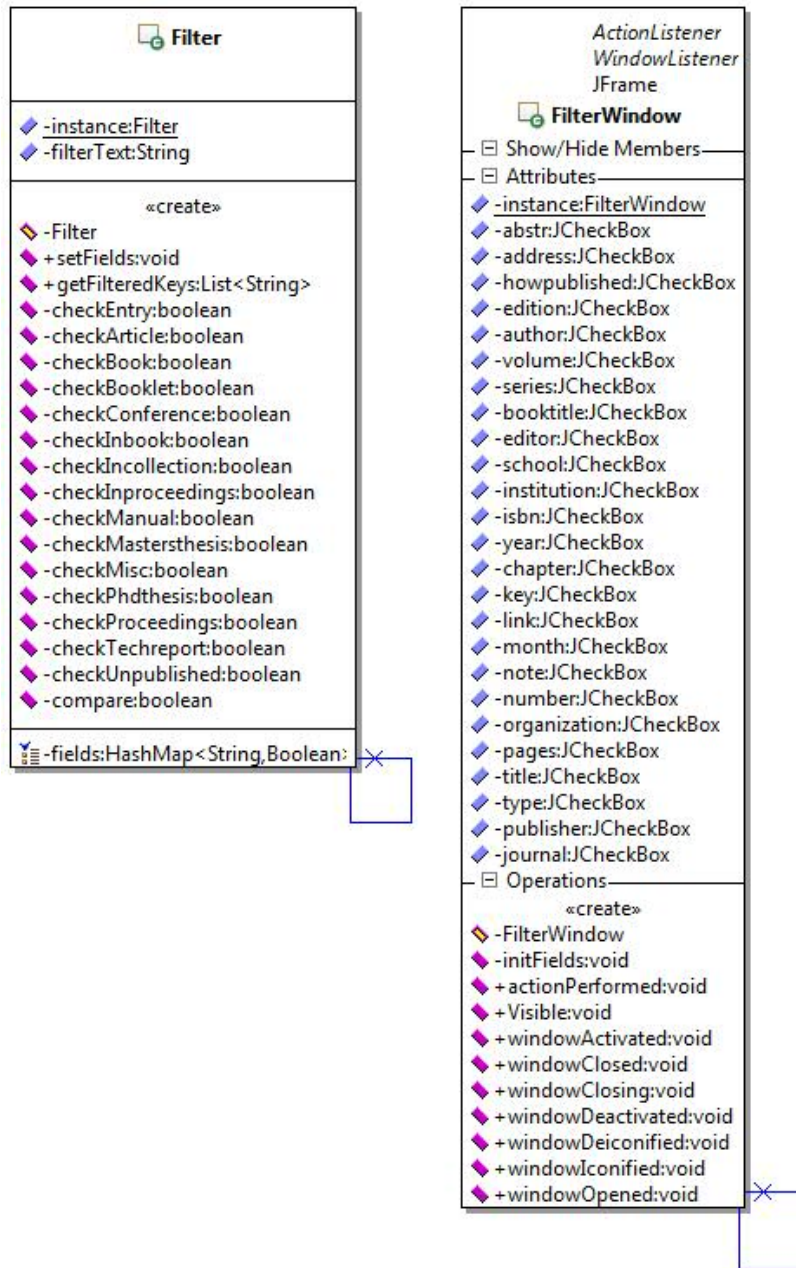


Abbildung 5: Klassendiagramm Package *de.BibTexMaker.GUI.Filter*

1.2.5.1 Klasse Filter

Die Klasse `Filter` stellt Methoden zur Filterung bereit. Dabei werden die Publikationen gemäß den Filtereinstellungen gefiltert. Entsprechend wird die Auswahlliste der Publikationen erstellt.

Die Methode `getFields` gibt die Liste der Felder zurück mit den Informationen welches Feld in den Filter einzubeziehen ist.

Die Klasse beinhaltet zwei Methoden `setFields`, welche sich in den Parametern und des beinhaltenden Algorithmus unterscheiden.

Mit der Methode `setFields` kann über den Parameter `fields` die komplette Liste der Felder die in den Filter einzubeziehen sind überschrieben werden.

Mit der Methode `setFields` kann über den Parameter `fields` Änderungen in der Liste der Felder welche in den Filter einzubeziehen sind vorgenommen werden.

Mit der Methode `getFilteredKeys` wird der aufrufenden Methode eine Liste der `BibKeys` übergeben, welche einen Treffer in der Datenbank auf den Filter angewandt erzielen. Dazu ist eine Liste (`allkeys`) mit `BibKeys` der Methode zu übergeben. Des Weiteren muss mittels des Parameters `filterText` der entsprechende Filtertext übergeben werden, nach dem die `BibKeys` in der Datenbank zu filtern sind.

Mit der Methode `checkEntry` wird geprüft, ob es eine Übereinstimmung mit dem Filtertext und der Publikation welche mit dem Parameter `bibkey` der Methode übergeben wird. Dabei werden nur die Felder berücksichtigt, welche auch in den Filter einzubeziehen sind.

Die Methoden `checkArticle`, `checkBook` usw. untersuchen die einzelnen Daten der Publikation, welche mit dem Parameter `bibkey` der Methode übergeben wurde, gemäß den Filtereinstellungen auf Übereinstimmung mit dem Filtertext.

Die Methode `compare` vergleicht den Filtertext mit dem Text der über den Parameter `content` der Methode übergeben wird. Bei Übereinstimmung gibt die Methode `true` zurück, ansonsten `false`.

1.2.5.2 Klasse `FilterWindow`

Die Klasse `FilterWindow` ist von `JFrame` abgeleitet und implementiert den `ActionListener` und den `WindowListener`. Die Klasse stellt das Fenster bereit in dem der Anwender die Felder auswählen kann, welche in die Filterung einzubeziehen sind.

Mit der Methode `initFields` werden die Checkboxes gemäß der Liste der einzubeziehenden Felder initialisiert.

Die Methode `actionPerformed` wird ausgeführt, wenn ein Klick auf eine Checkbox erfolgt. Die Methode ändert entsprechend die Liste der einzubeziehenden Felder im Filter ab.

Mit der Methode `Visible` kann eine Methode das Fenster öffnen. Die Methode stellt anhand der Bildschirm-auflösung die Größe des Fensters ein und positioniert es auf dem Bildschirm.

Die Methode *windowClosing* wird ausgeführt wenn ein *WindowClosingEvent* ausgelöst wird. Die Methode ruft dann die Methode *initFields* auf.

1.2.6 Package InsertWindow

Das Package *de.BibTexMaker.GUI.InsertWindow* stellt die Klassen für das Fenster zum Hinzufügen von Publikationen zur Verfügung.

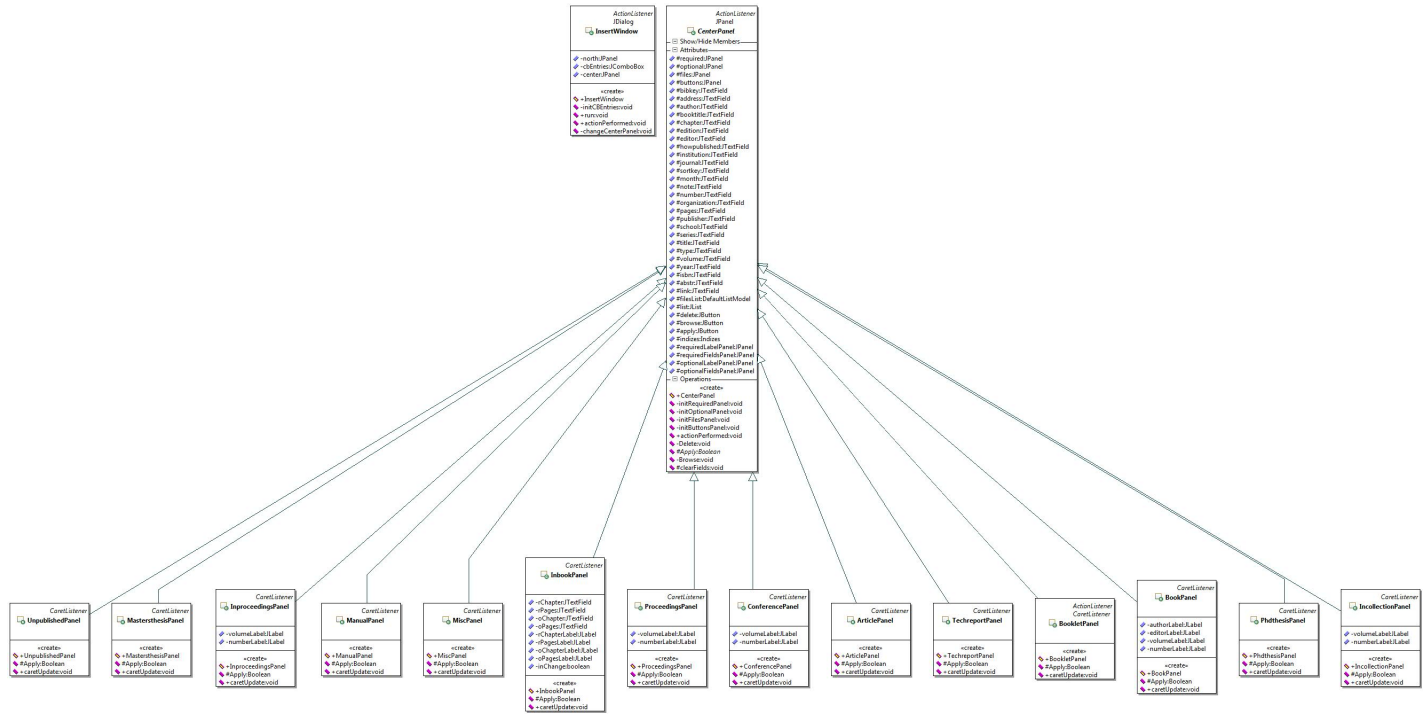


Abbildung 6: Klassendiagramm Package *de.BibTexMaker.GUI.InsertWindow*

1.2.6.1 Klasse InsertWindow

Die Klasse *InsertWindow* stellt der GUI ein Fenster zum Hinzufügen von Publikation zur Verfügung. Die Klasse ist von *JDialog* abgeleitet und implementiert den *ActionListener*.

Mit der Methode *run* wird das Fenster aufgerufen. Die Methode stellt anhand der Bildschirmauflösung die Größe des Fensters ein und positioniert es mittig auf dem Bildschirm.

Mit der Methode *initCBEntries* wird die Auswahlliste der Publikationstypen initialisiert.

Die Methode *actionPerformed* reagiert auf *ActionEvents*. Dabei prüft es ob das Kommando *comboBoxChanged* übergeben wurde. Ist dies der Fall wird der Wechsel des CenterPanels ausgelöst. Das Kommando *comboBoxChanged* wird ausgelöst, wenn in der Auswahlliste ein anderer Publikationstyp ausgewählt wird. Da der Publikationstyp gewechselt wurde, muss auch das entsprechende Panel in das CenterPanel geladen werden.

Mit der Methode *changeCenterPanel* wird das Panel in das CenterPanel geladen, dass zu dem Publikationstyp gehört, welche in der Auswahlliste ausgewählt wurde.

1.2.6.2 Klasse CenterPanel

Die Klasse *CenterPanel* ist von *JPanel* abgeleitet und implementiert den *ActionListener*. Die Klasse wird nicht direkt von *InsertWindow* verwendet, sondern dient für die jeweiligen CenterPanels (je Publikationstyp) als Oberklasse von der abgeleitet wird.

Die Methode *initRequiredPanel* initialisiert das Panel der Pflichtfelder.

Die Methode *initOptionalPanel* initialisiert das Panel der optionalen Felder.

Die Methode *initFilesPanel* initialisiert das Panel mit der Liste der BibTex-Dateien.

Die Methode *initButtonsPanel* initialisiert das Panel mit den Schaltflächen.

Die Methode *actionPerformed* reagiert auf *ActionEvents*. Entsprechend des übergebenen Kommandos, wird eine entsprechende Methode ausgeführt.

Wenn die Schaltfläche ENTFERNEN betätigt wird, wird die Methode *Delete* ausgeführt. Die Methode entfernt die ausgewählten BibTex-Dateien aus der Liste der der Publikation zugeordneten BibTex-Dateien.

Die Methode *Apply* ist eine abstrakte Methode und wird ausgeführt wenn die Schaltfläche SPEICHERN betätigt wurde. Der eigentliche Quellcode wird durch die ableitende Klasse gestellt.

Die Methode *Browse* öffnet ein Auswahlfenster mit denen man BibTex-Dateien der Liste hinzufügen kann. Sollte diese Datei noch nicht existieren, wird Sie zwar nicht angelegt, aber trotzdem in die Liste aufgenommen.

Mit der Methode *clearFields*, wird in allen Editierfeldern der Inhalt gelöscht und alle Einträge in der Liste der BibTex-Dateien entfernt.

1.2.6.3 abgeleitete Klassen von der Klasse CenterPanel

Zu den abgeleiteten Klassen von der Klasse *CenterPanel* gehören folgende Klassen:

- *ArticlePanel*
- *BookPanel*

- BookletPanel
- ConferencePanel
- InbookPanel
- IncollectionPanel
- InproceedingsPanel
- ManualPanel
- MastersthesisPanel
- MiscPanel
- PhdthesisPanel
- ProceedingsPanel
- TechreportPanel
- UnpublishedPanel

Alle zuvor aufgezählten Klassen besitzen den gleichen Aufbau. Sie unterscheiden sich lediglich darin, welche Pflichtfelder und optionalen Felder im Panel zur Anwendung kommen. Dies ist abhängig vom Publikationstyp der hinzuzufügenden Publikation. In der Liste kann man erkennen, dass der Publikationstyp sich im Namen der jeweiligen Klasse widerspiegelt. Im folgenden werden daher die Methode nicht explizit für jede Klasse vorgestellt, sondern allgemein.

Der Konstruktor initialisiert die Panels mit den Labeln und Editierfeldern der Pflicht- und optionalen Felder sowie der BibTex-Datei-Liste. Dem Konstruktor ist die Breite des Panels zu übergeben.

Die Methode *Apply* implementiert den Quellcode in die abstrakte Methode der Oberklasse *CenterPanel*. Die Methode wird ausgeführt, wenn die Schaltfläche SPEICHERN betätigt wurde. Die Methode liest die Daten aus den Editierfeldern aus und übergibt diese der Datenbank.

Die Methode *caretUpdate* prüft ob alle Pflichtfelder ausgefüllt sind. Ist dies der Fall so wird die Schaltfläche SPEICHERN freigegeben, ansonsten gesperrt. Die Methode wird aufgerufen, wenn ein *CaretEvent* ausgelöst wird. Der *CaretEvent* wird ausgelöst, wenn sich die Cursor-Position in einem der Editierfelder der Pflichtfelder ändert.

1.2.7 Package EditWindow

Das Package *de.BibTexMaker.GUI.EditWindow* stellt die Klassen für das Fenster zum Bearbeiten einer Publikation zur Verfügung.

1.2.7.2 Klasse `CenterPanel`

Die Klasse `CenterPanel` ist von `JPanel` abgeleitet und implementiert den `ActionListener`. Die Klasse wird nicht direkt von `EditWindow` verwendet, sondern dient für die jeweiligen `CenterPanel`s (je Publikationstyp) als Oberklasse von der abgeleitet wird.

Der Konstruktor initialisiert das `CenterPanel`. Dem Konstruktor ist der Index (bibkey) der zu bearbeitenden Publikation zu übergeben.

Die Methode `initRequiredPanel` initialisiert das Panel mit den Pflichtfelder. Dazu wird das Layout und die Border eingestellt.

Die Methode `initOptionalPanel` initialisiert das Panel mit den optionalen Feldern. Dazu wird das Layout und die Border eingestellt.

Die Methode `initFilesPanel` initialisiert das Panel mit den BibTex-Dateien.

Die Methode `initButtonPanel` initialisiert das Panel mit den Schaltflächen.

Die Methode `actionPerformed` reagiert auf `ActionEvents`. Dabei prüft die Methode welche Schaltfläche betätigt wurde. Entsprechend wird eine weitere Methode aufgerufen.

Wenn die Schaltfläche `ENTFERNEN` betätigt wird, wird die Methode `Delete` ausgeführt. Die Methode entfernt die ausgewählten BibTex-Dateien aus der Liste der der Publikation zugeordneten BibTex-Dateien.

Die Methode `Apply` ist eine abstrakte Methode und wird ausgeführt wenn die Schaltfläche `SPEICHERN` betätigt wurde. Der eigentliche Quellcode wird durch die ableitende Klasse gestellt.

Die Methode `Browse` öffnet ein Auswahlfenster mit denen man BibTex-Dateien der Liste hinzufügen kann. Sollte diese Datei noch nicht existieren, wird Sie zwar nicht angelegt, aber trotzdem in die Liste aufgenommen.

Die Methode `addEditPanelListener` dient anderen Klassen bzw. Methoden dazu sich an den `EditPanelListener` einzutragen, um bei entsprechenden Events informiert zu werden.

Die Methode `removeEditPanelListener` dient anderen Klassen bzw. Methoden dazu sich aus den `EditPanelListener` auszutragen.

Soll ein `EditPanelEvent` ausgelöst werden, so muss die Methode `notifyAdvertisement` aufgerufen werden. Diese geht dann die Liste der Klassen bzw. Methoden durch welche sich am `EditPanelListener` angemeldet haben und informiert diese, dass ein `EditPanelEvent` ausgelöst wurde.

1.2.7.3 abgeleitete Klassen von der Klasse `CenterPanel`

Zu den abgeleiteten Klassen von der Klasse `CenterPanel` gehören folgende Klassen:

- ArticlePanel
- BookletPanel
- BookPanel
- ConferencePanel
- InbookPanel
- IncollectionPanel
- InproceedingsPanel
- ManualPanel
- MastersthesisPanel
- MiscPanel
- PhdthesisPanel
- ProceedingsPanel
- TechreportPanel
- UnpublishedPanel

Alle zuvor aufgezählten Klassen besitzen den gleichen Aufbau. Sie unterscheiden sich lediglich darin, welche Pflichtfelder und optionalen Felder im Panel zur Anwendung kommen. Dies ist abhängig vom Publikationstyp der hinzuzufügenden Publikation. In der Liste kann man erkennen, dass der Publikationstyp sich im Namen der jeweiligen Klasse widerspiegelt. Im folgenden werden daher die Methode nicht explizit für jede Klasse vorgestellt, sondern allgemein.

Der Konstruktor befüllt die in CenterPanel initialisierten Panels mit den Labeln und Editierfeldern der Pflicht- und optionalen Felder sowie der BibTex-Datei-Liste. Dem Konstruktor ist der Index der Publikation und die Breite des Panels zu übergeben.

Die Methode *Apply* implementiert den Quellcode in die abstrakte Methode der Oberklasse *CenterPanel*. Die Methode wird ausgeführt, wenn die Schaltfläche SPEICHERN betätigt wurde. Die Methode liest die Daten aus den Editierfeldern aus und übergibt diese der Datenbank.

Die Methode *caretUpdate* prüft ob alle Pflichtfelder ausgefüllt sind. Ist dies der Fall so wird die Schaltfläche SPEICHERN freigegeben, ansonsten gesperrt. Die Methode wird aufgerufen, wenn ein *CaretEvent* ausgelöst wird. Der *CaretEvent* wird ausgelöst, wenn sich die Cursor-Position in einem der Pflichtfelder ändert.

1.2.7.4 Klasse `EditPanelListener`

Das Interface `EditPanelListener` wird von den Klassen eingebunden, welche auf `EditPanelEvents` hören. Es ist von `EventListener` abgeleitet und führt die Methode `advertisement` ein. Diese Methode wird in den entsprechenden Klassen ausgeführt, wenn ein `EditPanelEvent` ausgelöst wurde und die Klassen sich am Listener angemeldet haben.

1.2.7.5 Klasse `EditPanelEvent`

Die Klasse `EditPanelEvent`, wird dazu genutzt, um bei einem auslösen eines `EditPanelEvents` ein Objekt dieser Klasse zu erzeugen. Die Klasse ist von `EventObject` abgeleitet.

Die Methode `getEventCommand` gibt der aufrufenden Methode das Kommando des Events zurück.

1.2.8 Package `createWindow`

Das Package `de.BibTexMaker.GUI.createWindow` stellt eine Klasse mit entsprechenden Methoden der GUI zur Verfügung, welche ein Fenster erzeugen, dass eine Auswahl der BibTex-Dateien bereitstellt. Aus dieser Liste kann dann der Anwender eine BibTex-Datei auswählen, welche erstellt werden soll.

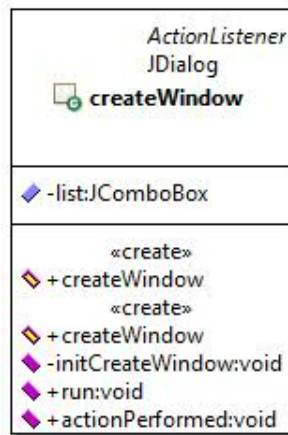


Abbildung 8: Klassendiagramm Package `de.BibTexMaker.GUI.createWindow`

1.2.8.1 Klasse `createWindow`

Die Klasse `createWindow` stellt einen Dialog zum erstellen einer BibTex-Datei bereit. Die Klasse ist von `JDialog` abgeleitet.

Ein Objekt der Klasse kann über zwei verschiedene Konstruktoren initialisiert werden.

Dem ersten Konstruktor sind keine Parameter zu übergeben. Er führt die Methode *initCreateWindow* aus, welche den Dialog initialisiert.

Dem zweiten Konstruktor ist mittels des Parameters *file* der Dateiname der BibTex-Datei zu übergeben, welche nach der Initialisierung in der Auswahlliste selektiert, also vorausgewählt, sein soll. Dazu führt der Konstruktor zunächst die Methode *initCreateWindow* aus, welche den Dialog initialisiert. Anschließend wird die entsprechende BibTex-Datei in der Auswahlliste selektiert.

Mit der Methode *initCreateWindow* wird der Dialog initialisiert.

Mit der Methode *run* wird der Dialog aufgerufen. Die Methode stellt dabei die Größe des Dialogfensters ein und positioniert das Fenster auf dem Bildschirm.

Die Methode *actionPerformed* reagiert auf *ActionEvents*. Die Methode prüft dann ob die Schaltfläche ERSTELLEN betätigt wurde. Ist dies der Fall wird der Serialisierer aufgerufen und die BibTex-Datei übergeben, welche erzeugt werden soll. Zum Schluss wird der createWindow-Dialog geschlossen.

1.3 Package de.BibTexMaker.Database

Das Package *de.BibTexMaker.Database* beinhaltet alle Klassen und Packages der Datenbank.

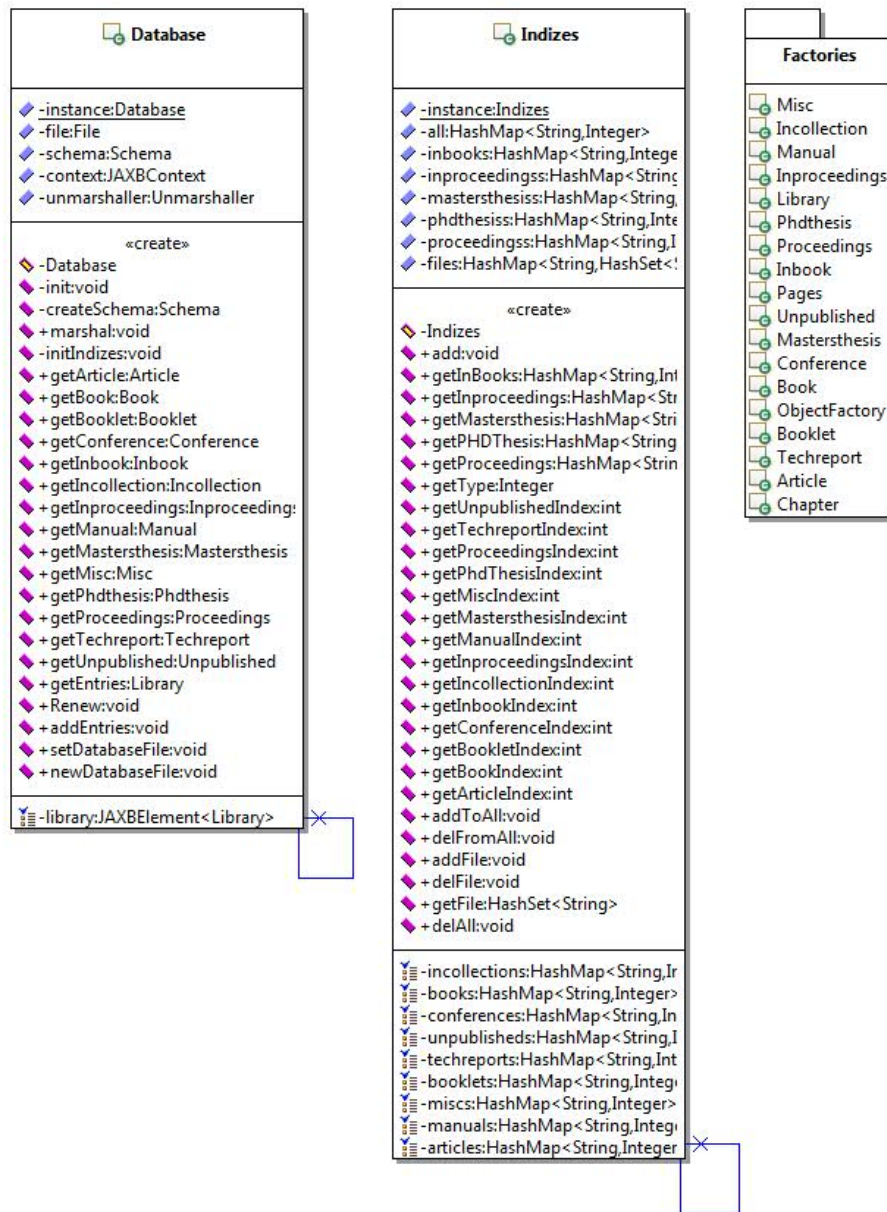


Abbildung 9: Klassendiagramm Package *de.BibTexMaker.Database*

1.3.1 Klasse Database

Die Klasse *Database* nutzt die Programmschnittstelle *Java Architecture for XML Binding (JAXB)* in der Version 2.1. Damit wird ermöglicht Daten aus einer XML-Datei automatisch an Java-Klassen zu binden und aus einem XML-Schema heraus zu generieren. Weiterhin stellt die Datenbank Methoden zum Marshallen und Unmarshallen zur Verfügung. Sowohl beim Marshallen als auch beim Unmarshallen werden die Daten validiert und auf ihre Korrektheit und Konsistenz geprüft. Es werden auch Getter und Setter Methoden bereit gestellt, um Daten in die Datenbank zu schreiben bzw. aus der Datenbank zu lesen. Für die diversen Zugriffe auf die Daten werden Indizierungen bereitgestellt. Besser gesagt die Indizierungen werden hier erstellt und über die Klasse *Indizes* können diese abgefragt werden.

Die Klasse ist nach dem Prinzip des Singleton-Pattern geschrieben. Dies dient dazu sicherzustellen, dass innerhalb des Programms nur eine Instanz der Klasse existiert, so dass immer auf die Richtige Instanz zugegriffen wird. Der Zugriff erfolgt über die Methode *getInstance*. Diese prüft ob bereits eine Instanz der Klasse angelegt wurde und legt eine an, wenn dies nicht der Fall ist. Anschließend gibt die Methode die Instanz an die aufrufende Methode zurück.

Über die Methode *init* wird der Datenbank mittels des Parameters *file* die Datenbankdatei (XML-Datei) übergeben. Daraufhin wird JAXB initialisiert und die Daten der XML-Datei an die entsprechenden Java-Klassen aus dem Package *Factories* gebunden.

Die Methode *Schema* lädt das XML-Schema für die BibTexMaker-Datenbank und gibt dies der aufrufenden Methode zurück.

Die Methode *marshal* wird aufgerufen, um die Daten aus den Java Objekten in eine XML-Datei zu generieren.

Die Methode *initIndizes* legt Listen der einzelnen Publikationen zu den entsprechenden Publikationstypen an. Diese Listen werden z.B. von der GUI für die Auswahllisten der Publikationen verwendet.

Die verschiedenen Getter-Methode zu den Publikationstypen (z.B. *getArticle*, *getBook* usw.) geben die Daten zu einer über den Parameter *bibkey* bestimmten Publikation an die aufrufende Methode zurück.

Über die Methode *getLibrary* kann die gesamte Datenbank abgefragt werden.

Die Methode *getEntries* übergibt der aufrufenden Methode alle Datensätze der Datenbank.

Mit der Methode *Renew* kann eine externe Methode veranlassen, dass die Indizierungen neu erstellt werden.

Um Publikationen in die Datenbank aufzunehmen, muss die Methode *addEntries* aufgerufen werden. Über den Parameter *entries* sind dabei die Datensätze der Methode zu übergeben.

Soll die Datenbank gewechselt werden, dann ist die Methode *setDatabaseFile* aufzurufen. Über dem Parameter *file* ist die zu ladende Datenbankdatei zu übergeben.

Mithilfe der Methode *newDatabaseFile* kann eine neue Datenbankdatei angelegt werden. Der Methode ist über den Parameter *file* die Datenbankdatei zu übergeben.

1.3.1.1 Klasse *Indizes*

Die Klasse *Indizes* beinhaltet Methode um Indizierungslisten zu erstellen und zu bearbeiten. Die Indizierungslisten werden von der GUI verwendet, um damit die Auswahlliste der Publikationen zu befüllen. Die Klasse wurde nach dem Prinzip des Singleton-Pattern geschrieben, um sicherzustellen, dass immer auf die aktuellen Indizierungslisten zugegriffen wird.

Die Methode *getInstance* übergibt die einzige Instanz der Klasse *Indizes* an die aufrufende Methode. Dabei wird vorher geprüft ob die Instanz schon angelegt wurde. Ist dies nicht der Fall, so wird dies noch erledigt.

Über die Methode *add* können einer Indizierungsliste eines bestimmten Publikationstypen weitere Einträge hinzugefügt werden.

Die Methode *getAll* gibt der aufrufenden Methode die Indizierungsliste aller Publikationstypen zurück.

Die Getter-Methode zu den Publikationstypen (z.B. *getArticles*, *getBooks*, usw.) geben der aufrufenden Methode die Indizierungsliste des entsprechenden Publikationstypen zurück.

Mit der Methode *getType* kann aus der Indizierungsliste aller Publikationen der Publikationstyp extrahiert werden. Als Parameter muss der entsprechende BibKey der Publikation übergeben werden.

Die Getter-Methoden zu den Publikationen (z.B. *getTechreportIndex*, *getUnpublishedIndex*, usw.) geben der aufrufenden Methode den Index einer bestimmten Publikation zurück. Die Publikation wird über den Parameter *bibkey* referenziert.

Mit der Methode *addToAll* kann der Indizierungsliste aller Publikationen ein neuer Eintrag hinzugefügt werden.

Mit der Methode *delFromAll* kann ein Eintrag aus der Indizierungsliste aller Publikationen entfernt werden.

Mit der Methode *addFile* kann der Liste der Zuordnungen von BibTex-Dateien und Publikationen ein neuer Eintrag hinzugefügt werden.

Mit der Methode *delFile* kann aus der Liste der Zuordnungen von BibTex-Dateien und Publikationen ein Eintrag entfernt werden.

Die Methode *getFile* gibt der aufrufenden Methode eine Liste der Publikationen zurück, welche einer bestimmten BibTex-Datei zugeordnet sind. Die Bibtex-Datei wird mit dem Parameter *filename* der Methode übergeben.

Die Methode *getFiles* gibt eine Liste aller in der Datenbank eingetragenen BibTex-Dateien zurück.

Mit der Methode *delAll* können alle Einträge aus allen Indizierungslisten entfernt werden.

1.3.1.2 Packages *de.BibTexMAker.Database.Factories*

Das Package *Factories* enthält die Klassen, welche die Java-Objekte der XML-Elemente entsprechen. Sie dienen

als Schnittstelle zwischen XML und Java.

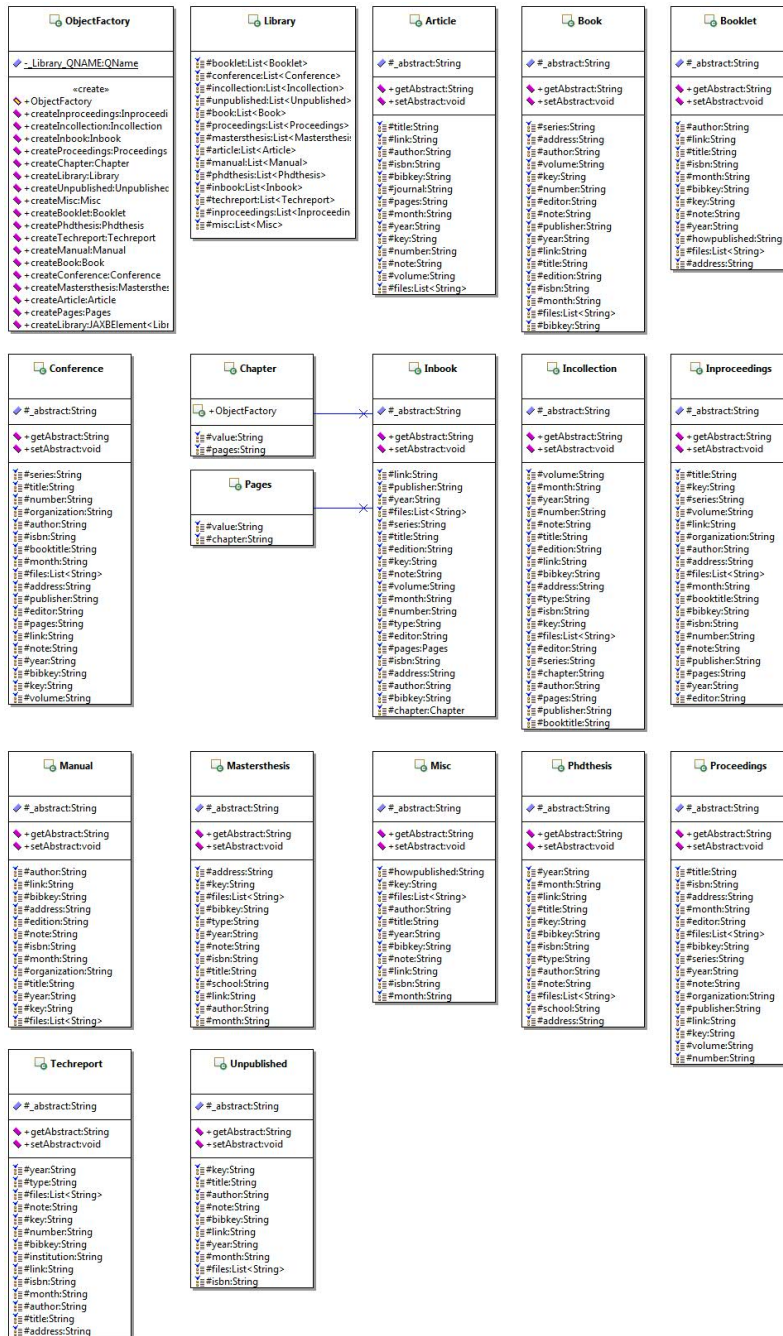


Abbildung 10: Klassendiagramm Package *de.BibTexMaker.Database.Factories*

Die Klassen sind nach dem Prinzip des Factory Pattern geschrieben und wurden über das Framework JAXB automatisch erstellt. Dazu wurde eine XML-Schemata-Datei erstellt und diese dem ANT Compiler mittels einer build.xml-Datei übergeben. In der build.xml stehen die Verarbeitungsregeln für den ANT Compiler und in der XML-Schemata-Datei (library.xsd) die Regeln zu den XML-Elementen. Aus diesen Dateien und deren darin befindlichen Angaben hat der ANT Compiler die im Package *de.BibTexMAker.Database.Factorie* enthaltenen Klassen erstellt.

Aufgrund der automatischen Erzeugung der Klassen und der Nutzung des Frameworks JAXB wird hier auf eine Dokumentation der einzelnen Klassen verzichtet und auf die Kommentare in den Klassen bzw. den diversen Dokumentationen zu JAXB verwiesen.

1.4 Package de.BibTexMaker.BibTex

1.4.1 Klasse Parser

1.4.2 Klasse ParserException

1.4.3 Klasse Serializer

1.4.4 Klasse Validator

1.5 Package de.BibTexMaker.Utiles

1.5.1 Klasse Fields

1.5.2 Klasse BibTexFileFilter

1.5.3 Klasse EntryTypes