Architekturdokument

autoPSI

Gruppe 06: Feledi, Firato, Mildner, Zapotocky

Inhalt

Systemarchitektur

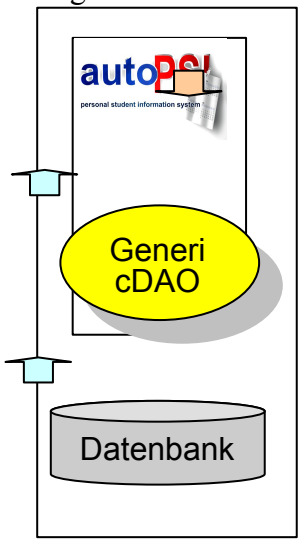
UserInterface

autoPSI soll über eine übersichtliche und intuitiv bedienbare Benutzeroberfläche verfügen. Alle Funktionen sollen schnell erreichbar sein. Es sollen möglichst wenige und vor allem selbsterklärende Steuerelemente zum Einsatz kommen, denn es soll vermieden werden, dass Benutzer in einer Dokumentation nachschlagen müssen. Ähnliche Objekte sollen in ähnlichen Abläufen bearbeitbar sein, da dies im allgemeinen sehr intuitiv ist (Man probiert etwas, bemerkt,dass es funktioniert und kann den Ablauf dann an verschiedenen anderen Stellen im Programm wiederverwenden).
Durch möglichst sinnvolle Gliederung der Informationen in graphischer Form soll die Übersicht auch bei vielen Terminen möglichst erhalten bleiben.
Durch die Möglichkeit, mehrere Fenster zum Bearbeiten zu öffnen, wird das gleichzeitige Bearbeiten von mehreren Objekten ermöglicht (und dadurch auch Copy-And-Paste gefördert).

GenericDAO

Grundsätzliche Idee

Die grundsätzliche Idee ist es, den Datenbankzugriff so generisch wie möglich zu gestalten. Das zu erreichende Ziel ist es, dass möglichst nur ein Objekt tatsächlich mit der Datenbank kommuniziert und dass dieses eine Objekt auf beliebige Datenbanktabellen zugreifen kann und Daten auch in Form von Objekten zurückgeben kann.



Eingesetzte Techniken

Erreicht wird dies durch Verwendung von Reflection. Durch diese Technik ist es möglich, die Daten einer Datenbanktabelle in ein dem GenericDAO-Objekt unbekanntes Objekt zu schreiben.
Um dem Benutzer möglichst große Flexibilität beim Auffinden von Objekten zu geben, soll die Suche nach Objekten mithilfe eines sogenannten LookupObjects erfolgen. Dieses Objekt ist ein Template, welcher der Suchfunktion übergeben wird und bei dem diejenigen Eigenschaften, die gefundene Objekte aufweisen sollen, gesetzt sind, während alle anderen Felder Nullreferenzen sind.

JavaSpace

Die Idee hinter JavaSpaces

Die Idee hinter der JavaSpaces-Technologie ist es, einen möglichst einfach verwendbaren Datenspeicher für Objekte anzubieten. Ein JavaSpace bietet vier grundsätzliche Operationen:

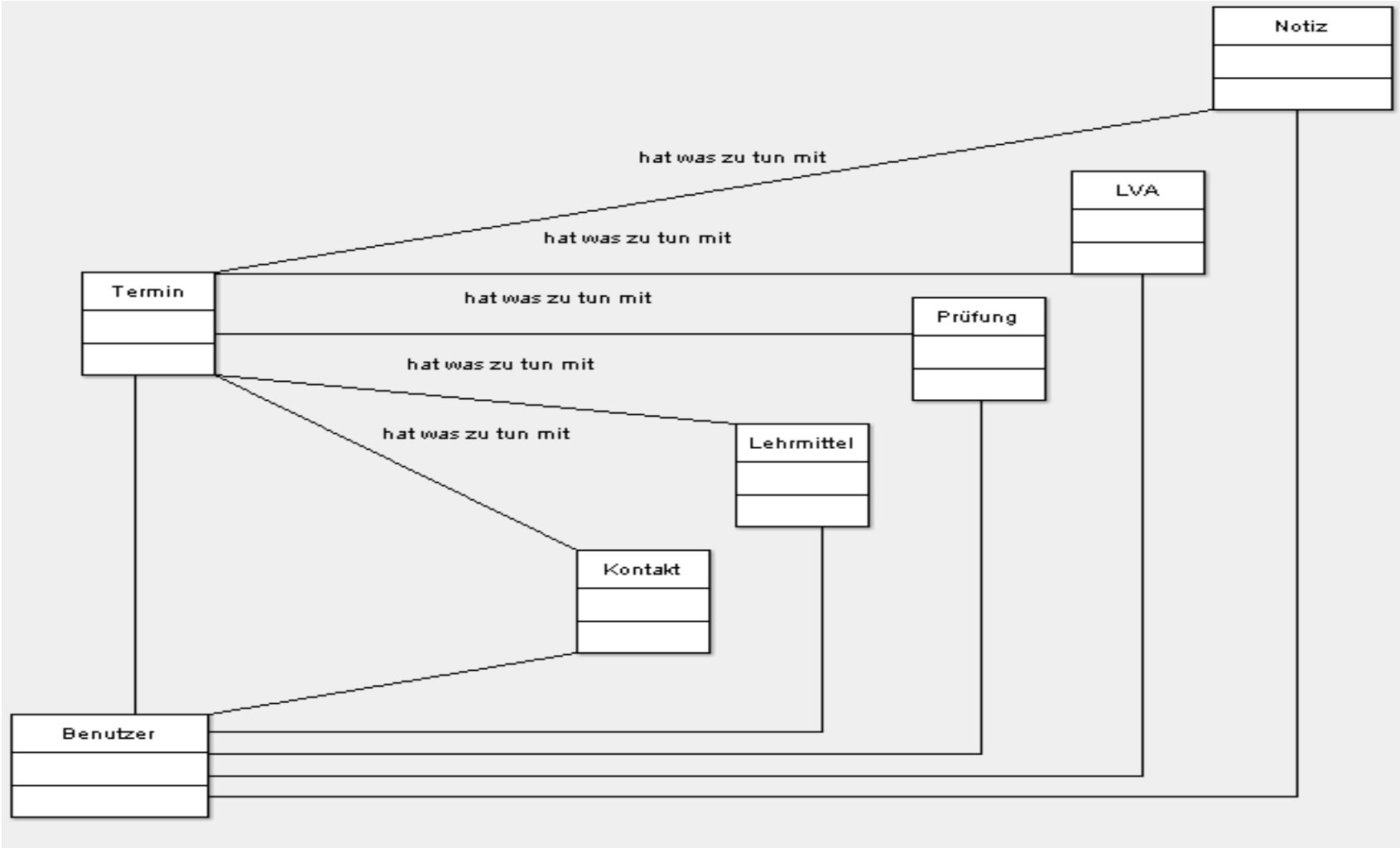
1. **Write**: Schreibt ein Objekt in den JavaSpace
2. **Read**: Liest ein Objekt aus dem JavaSpace
3. **Take**: Liest ein Objekt konsumierend aus dem JavaSpace
4. **Notify**: Informiert den Aufrufer, wenn ein Objekt in den JavaSpace eingefügt wurde, das der vom Aufrufer geforderten Beschreibung entspricht

Mit diesen Grundoperationen kann dann eine beliebige Business-Logik aufgebaut werden.

JavaSpaces in autoPSI

In autoPSI wird der Datenaustausch zwischen Studenten per JavaSpace unterstützt. Dadurch können Informationen einfach und effizient weitergegeben werden.

Domänenmodell



Pakete

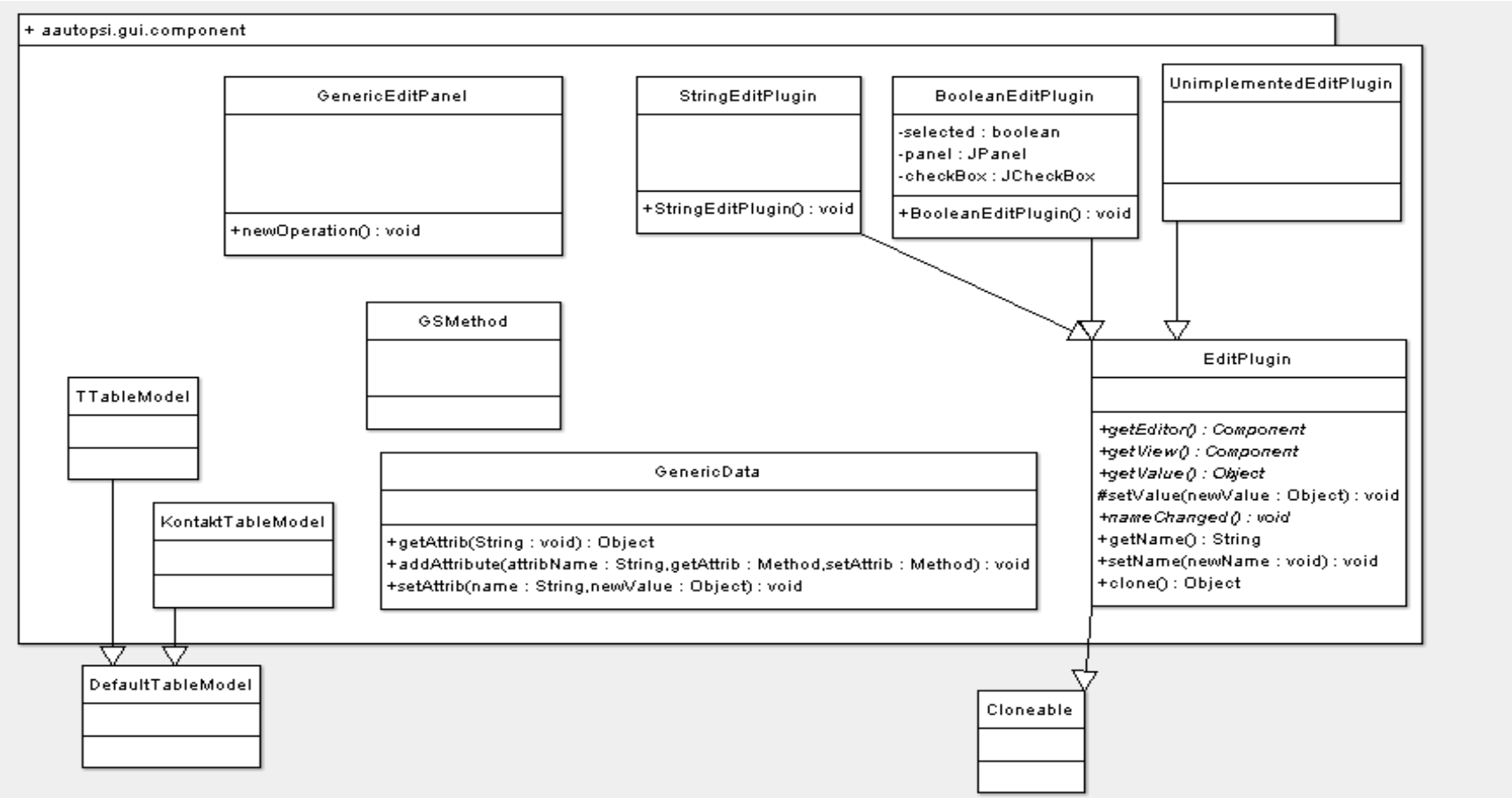
Paket-Diagramm

Paketbeschreibung

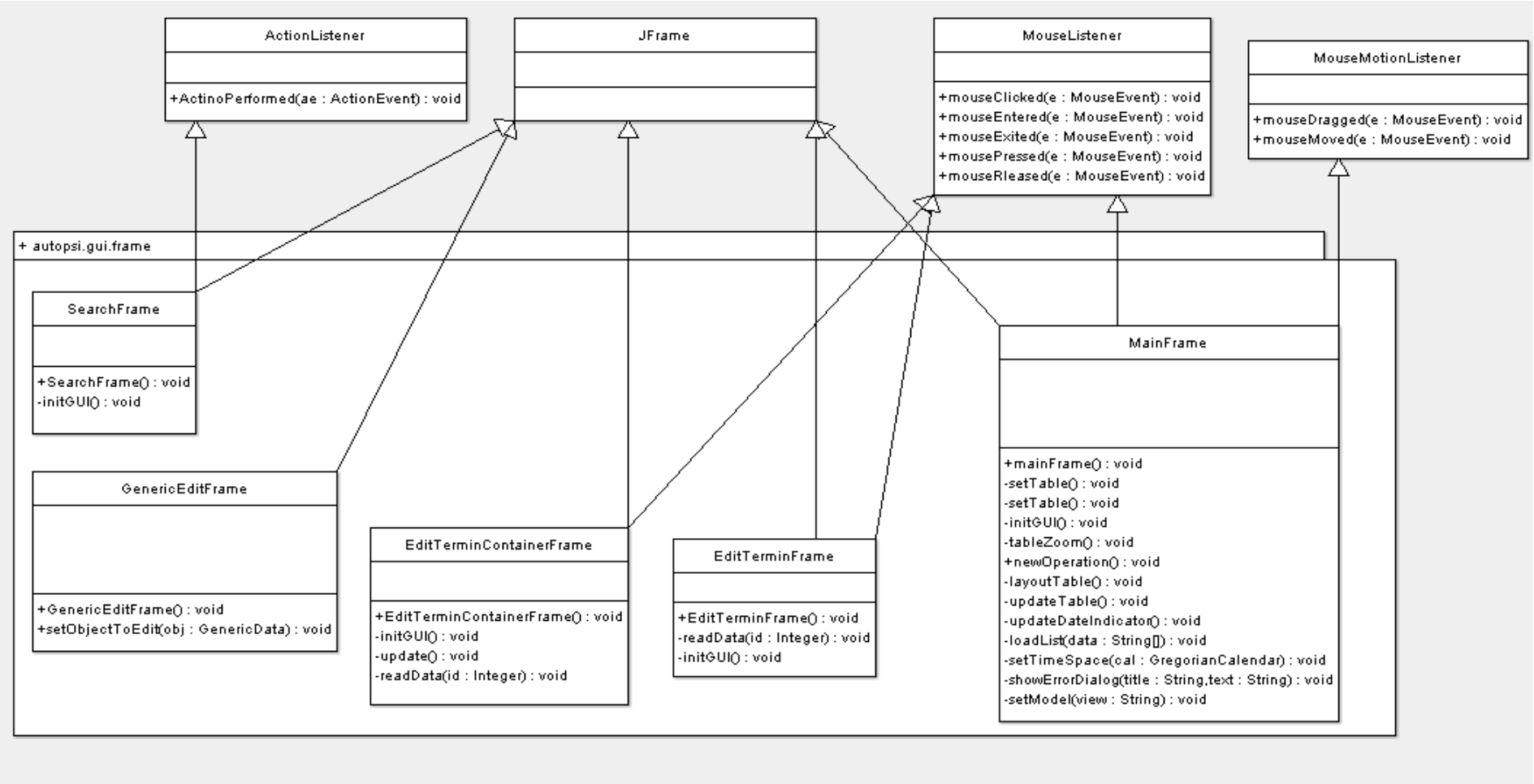
Package	*Funktionalität*
gui	Kurzbeschreibung: Enthält alle Klassen, die das graphische User Interface erzeugen Beschreibung: Enthält die Klassen MainFrame, TerminEditFrame, TermincontainerEditFrame, SearchFrame, GenericEditFrame
gui.frame	Kurzbeschreibung: enthält die von JFrame abgeleiteten Fenster des UserInterfaces Beschreibung: Implementierung des UserInterfaces
gui.components	Kurzbeschreibung: enthält selbst erstellte Komponenten, die beim UserInterface zum Einsatz kommen Beschreibung: Dieses Package enthält z.B. die TableModels für die Monatsansicht und die Ergebnistabelle im Suchfenster
gui.exceptions	
database	Kurzbeschreibung: Enthält alle Klassen, die mit der Datenbank oder dem Datenbankzugriff zu tun haben Beschreibung:
database.dao	Kurzbeschreibung: Enthält die Komponenten für den generischen Datenbankzugriff und das Objekt, welches benutzt wird, um eigene Datenbankobjekte abzuleiten Beschreibung: Enthält IGenericDAO, GenericDAO, GenericDataObject
database.exceptions	Kurzbeschreibung: Exceptions, die in Datenbankkomponente auftreten können Beschreibung: Enthält EAttributeNotFound, EDatabase, EDatabaseConnection
database.table	Kurzbeschreibung: Enthält Objekte, die die Daten halten Beschreibung:
database.sql	Kurzbeschreibung: Die SQL-Struktur, die das GenericDAO benutzt, um mit SQL-Abfragen umzugehen Beschreibung:
javaspace	Kurzbeschreibung: Enthält die Komponenten, die mit dem JavaSpace kommunizieren Beschreibung:
javaspace.exceptions	Kurzbeschreibung: Exceptions, die im Zusammenhang mit den JavaSpace-Objekten auftreten können Beschreibung:

Klassendiagramme

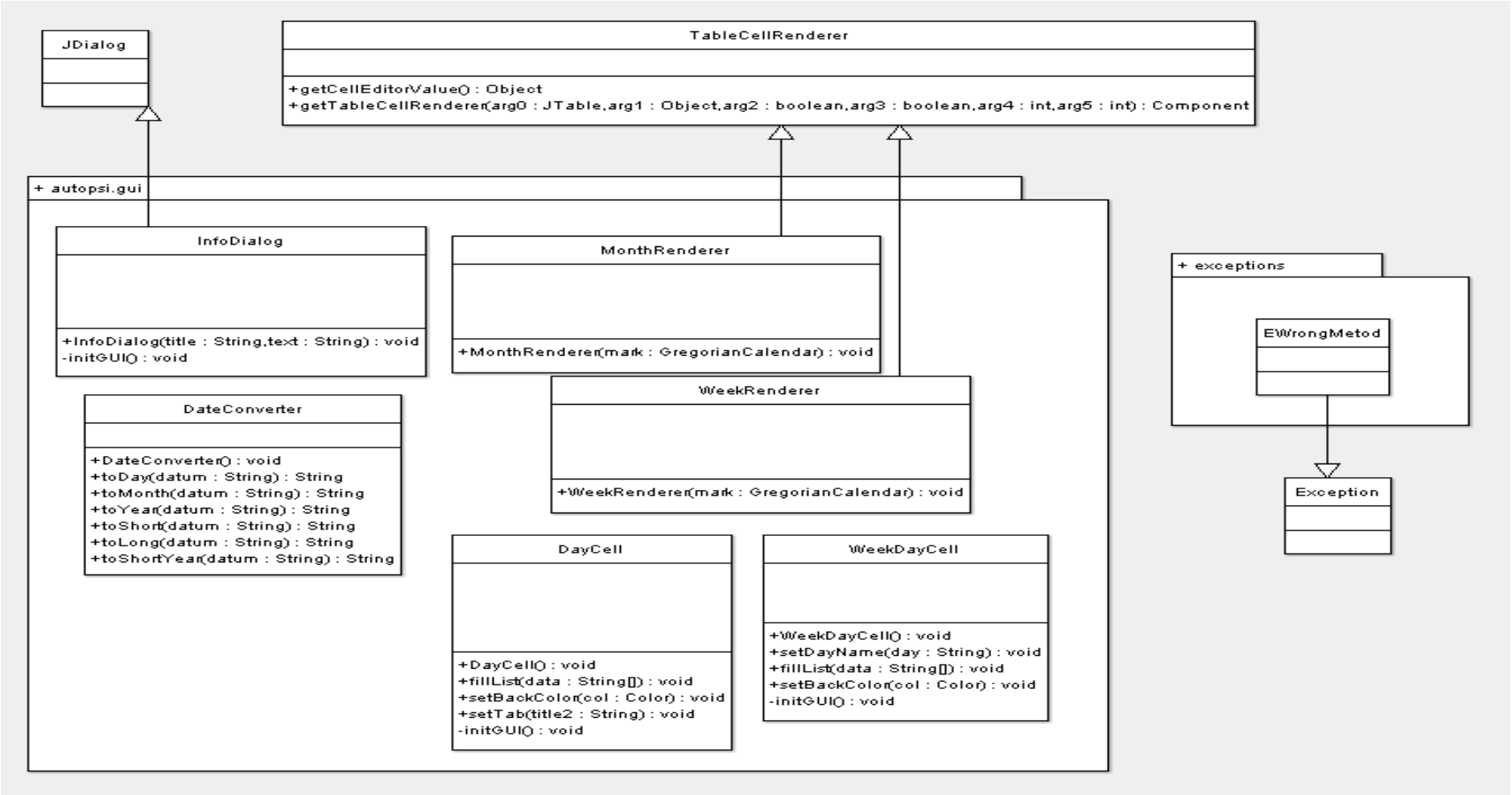
GUI – Komponenten



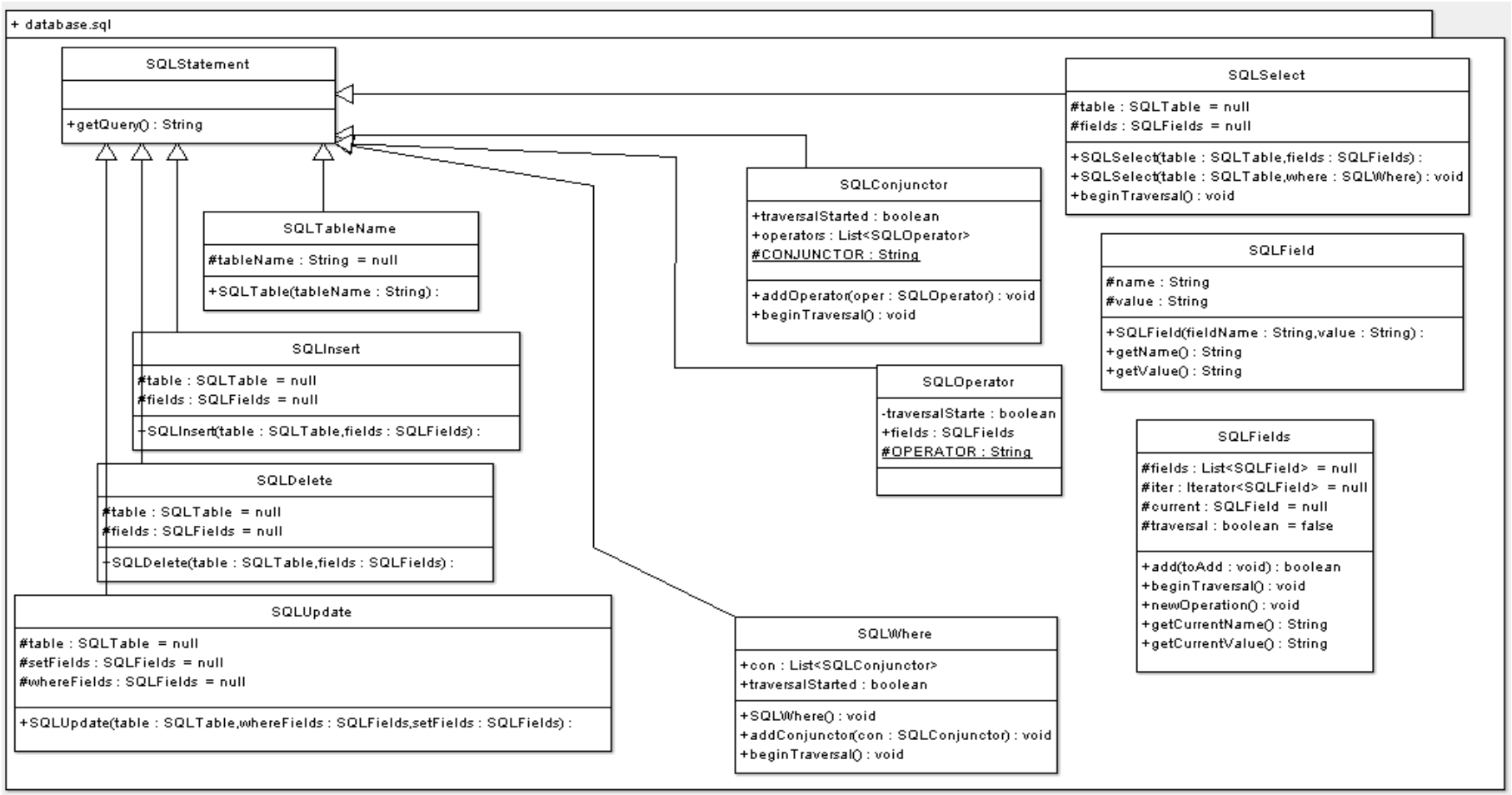
GUI – Frames


ActionListener
+ActinoPerformed(ae : ActionEvent) : void
JFrame
MouseListener
+mouseClicked(e : MouseEvent) : void
+mouseEntered(e : MouseEvent) : void
+mouseExited(e : MouseEvent) : void
+mousePressed(e : MouseEvent) : void
+mouseRleased(e : MouseEvent) : void
MouseMotionListener
+mouseDragged(e : MouseEvent) : void
+mouseMoved(e : MouseEvent) : void
+ autopsi.gui.frame
SearchFrame
+SearchFrame() : void
-initGUI() : void
GenericEditFrame
+GenericEditFrame() : void
+setObjectToEdit(obj : GenericData) : void
EditTerminContainerFrame
+EditTerminContainerFrame() : void
-initGUI() : void
-update() : void
-readData(id : Integer) : void
EditTerminFrame
+EditTerminFrame() : void
-readData(id : Integer) : void
-initGUI() : void
MainFrame
+mainFrame() : void
-setTable() : void
-setTable() : void
-initGUI() : void
-tableZoom() : void
+newOperation() : void
-layoutTable() : void
-updateTable() : void
-updateDateIndicator() : void
-loadList(data : String[]) : void
-setTimeSpace(cal : GregorianCalendar) : void
-showErrorDialog(title : String,text : String) : void
-setModel(view : String) : void

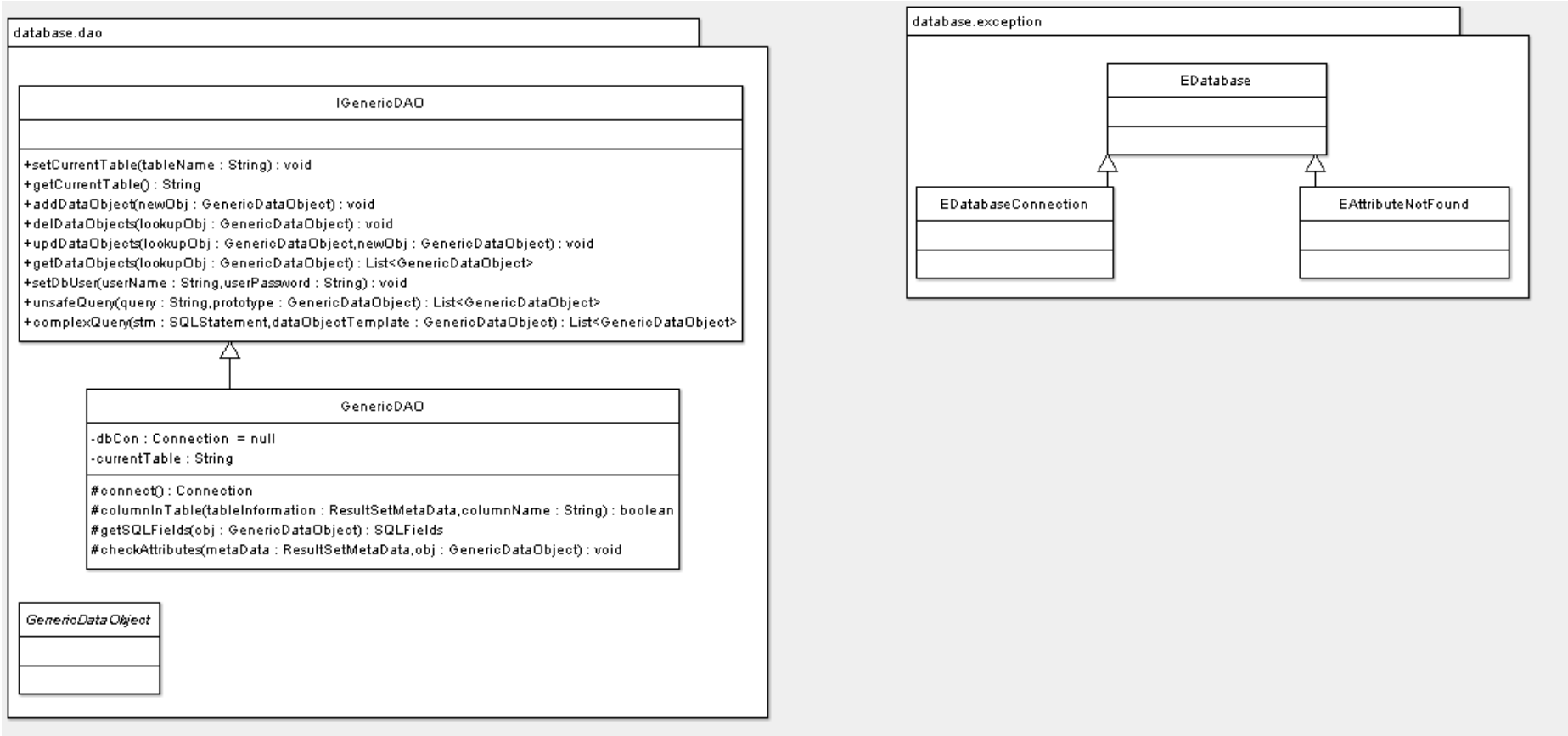
GUI


JDialog
TableCellRenderer
+getCellEditorValue() : Object
+getTableCellRenderer(arg0 : JTable,arg1 : Object,arg2 : boolean,arg3 : boolean,arg4 : int,arg5 : int) : Component
+ autopsi.gui
InfoDialog
+InfoDialog(title : String,text : String) : void
-initGUI() : void
MonthRenderer
+MonthRenderer(mark : GregorianCalendar) : void
WeekRenderer
+WeekRenderer(mark : GregorianCalendar) : void
DateConverter
+DateConverter() : void
+toDay(datum : String) : String
+toMonth(datum : String) : String
+toYear(datum : String) : String
+toShort(datum : String) : String
+toLong(datum : String) : String
+toShortYear(datum : String) : String
DayCell
+DayCell() : void
+fillList(data : String[]) : void
+setBackColor(col : Color) : void
+setTab(title2 : String) : void
-initGUI() : void
WeekDayCell
+WeekDayCell() : void
+setDayName(day : String) : void
+fillList(data : String[]) : void
+setBackColor(col : Color) : void
-initGUI() : void
+ exceptions
EWrongMetod
Exception

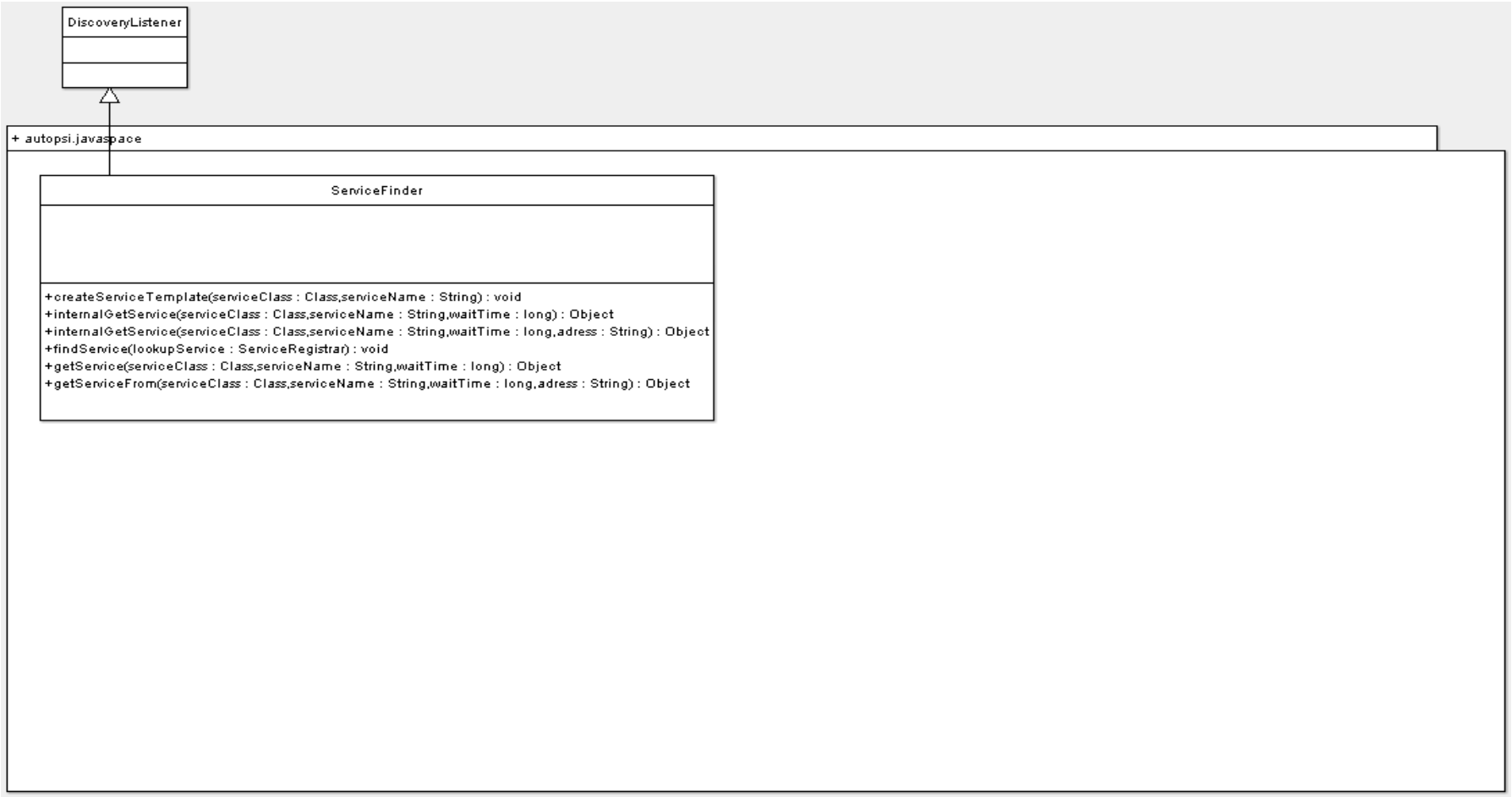
DAO - SQL


+ database.sql
SQLStatement
+getQuery() : String
SQLSelect
#table : SQLTable = null
#fields : SQLFields = null
+SQLSelect(table : SQLTable,fields : SQLFields) :
+SQLSelect(table : SQLTable,where : SQLWhere) : void
+beginTraversal() : void
SQLConjunctor
+traversalStarted : boolean
+operators : List<SQLOperator>
#CONJUNCTOR : String
+addOperator(oper : SQLOperator) : void
+beginTraversal() : void
SQLTableName
#tableName : String = null
+SQLTable(tableName : String) :
SQLField
#name : String
#value : String
+SQLField(fieldName : String,value : String) :
+getName() : String
+getValue() : String
SQLInsert
#table : SQLTable = null
#fields : SQLFields = null
+SQLInsert(table : SQLTable,fields : SQLFields) :
SQLOperator
-traversalStarte : boolean
+fields : SQLFields
#OPERATOR : String
SQLFields
#fields : List<SQLField> = null
#iter : Iterator<SQLField> = null
#current : SQLField = null
#traversal : boolean = false
+add(toAdd : void) : boolean
+beginTraversal() : void
+newOperation() : void
+getCurrentName() : String
+getCurrentValue() : String
SQLDelete
#table : SQLTable = null
#fields : SQLFields = null
+SQLDelete(table : SQLTable,fields : SQLFields) :
SQLUpdate
#table : SQLTable = null
#setFields : SQLFields = null
#whereFields : SQLFields = null
+SQLUpdate(table : SQLTable,whereFields : SQLFields,setFields : SQLFields) :
SQLWhere
+con : List<SQLConjunctor>
+traversalStarted : boolean
+SQLWhere() : void
+addConjunctor(con : SQLConjunctor) : void
+beginTraversal() : void

DAO

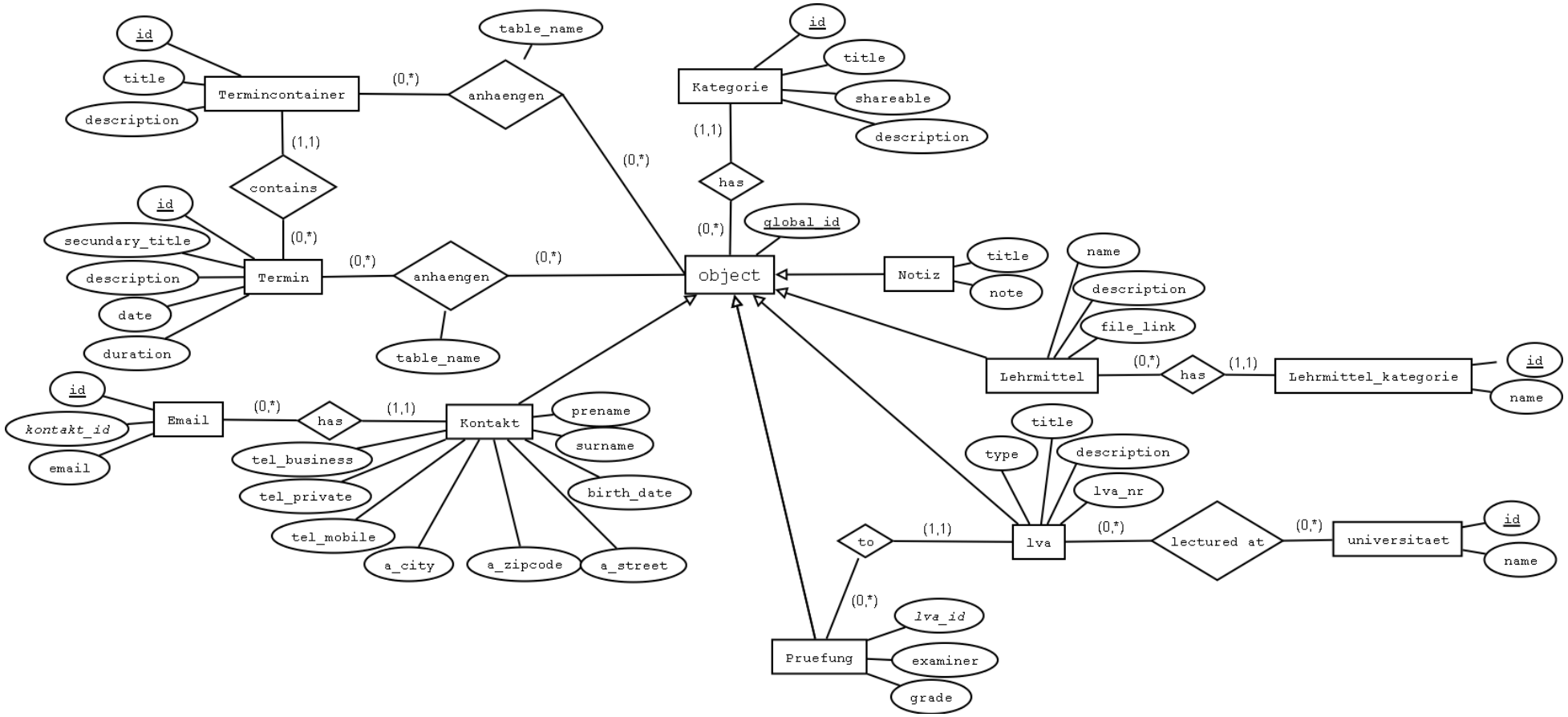

database.dao
IGenericDAO
+setCurrentTable(tableName : String) : void
+getCurrentTable() : String
+addDataObject(newObj : GenericDataObject) : void
+delDataObjects(lookupObj : GenericDataObject) : void
+updDataObjects(lookupObj : GenericDataObject,newObj : GenericDataObject) : void
+getDataObjects(lookupObj : GenericDataObject) : List<GenericDataObject>
+setDbUser(userName : String,userPassword : String) : void
+unsafeQuery(query : String,prototype : GenericDataObject) : List<GenericDataObject>
+complexQuery(stm : SQLStatement,dataObjectTemplate : GenericDataObject) : List<GenericDataObject>
GenericDAO
-dbCon : Connection = null
-currentTable : String
#connect() : Connection
#columnInTable(tableInformation : ResultSetMetaData,columnName : String) : boolean
#getSQLFields(obj : GenericDataObject) : SQLFields
#checkAttributes(metaData : ResultSetMetaData,obj : GenericDataObject) : void
GenericDataObject
database.exception
EDatabase
EDatabaseConnection
EAttributeNotFound

SBC


DiscoveryListener
+ autopsi.javaspace
ServiceFinder
+createServiceTemplate(serviceClass : Class,serviceName : String) : void
+internalGetService(serviceClass : Class,serviceName : String,waitTime : long) : Object
+internalGetService(serviceClass : Class,serviceName : String,waitTime : long,adress : String) : Object
+findService(lookupService : ServiceRegistrar) : void
+getService(serviceClass : Class,serviceName : String,waitTime : long) : Object
+getServiceFrom(serviceClass : Class,serviceName : String,waitTime : long,adress : String) : Object

Datenbankbeschreibung

ER-Diagramm



Datenbanktabellen

object

Feldname	*Typ*	*Optionen*	*Bemerkung*
global_id	INTEGER	!1+P	Globale ID, wird von einer Funktion in einer Helper-Klasse erzeugt und ist für **alle anhängbaren Objekte eindeutig**

termincontainer

Feldname	*Typ*	*Optionen*	*Bemerkung*
id	INTEGER	!1+P	
title	VARCHAR(255)		
description	VARCHAR(255)		

termin

Feldname	*Typ*	*Optionen*	*Bemerkung*
id	INTEGER	!1+P	
termin_kategorie_id			Fremdschlüssel zu (Termin_Kategorie.id)
secondary_title	VARCHAR(255)		
description	VARCHAR(255)		
date	DATE		Anfangsdatum- und Zeit des Termins
duration	INTEGER		Termindauer in Minuten
place	VARCHAR(255)		

lva

Feldname	*Typ*	*Optionen*	*Bemerkung*
global_id	INTEGER	!1+P	Globale ID, wird von einer Funktion in einer Helper-Klasse erzeugt und ist für **alle anhängbaren Objekte eindeutig**
kategorie_id	INTEGER	F	Key der kategorie; Relation has wurde in Object hereingezogen, da es keine table object gibt wird dieses attribut weitervererbt (object, kategorie)
title	VARCHAR(255)		
type	VARCHAR(255)		Auswahlliste
description	VARCHAR(255)		
lva_nr	VARCHAR(255)	!1	
uni_id	INTEGER	F	

Kontakt

Feldname	*Typ*	*Optionen*	*Bemerkung*
global_id	INTEGER	!1+P	Globale ID, wird von einer Funktion in einer Helper-Klasse erzeugt und ist für **alle anhängbaren Objekte eindeutig**
kategorie_id	INTEGER	F	Key der kategorie; Relation has wurde in Object hereingezogen, da es keine table object gibt wird dieses attribut weitervererbt (object, kategorie)
prename	VARCHAR(255)	!	
surname	VARCHAR(255)	!	
birth_date	DATE		
tel_private	VARCHAR(255)		
tel_business	VARCHAR(255)		
tel_mobile	VARCHAR(255)		
a_zipcode	INTEGER		
a_city	VARCHAR(255)		
a_adress	VARCHAR(255)		

Lehrmittel

Feldname	*Typ*	*Optionen*	*Bemerkung*
global_id	INTEGER	!1+P	Globale ID, wird von einer Funktion in einer Helper-Klasse erzeugt und ist für **alle anhängbaren Objekte eindeutig**
kategorie_id	INTEGER	F	Key der kategorie; Relation has wurde in Object hereingezogen, da es keine table object gibt wird dieses attribut weitervererbt (object, kategorie)
lehrmittel_kategorie_id	INTEGER	F	
name	VARCHAR(255)		
description	VARCHAR(255)		
file_link	VARCHAR(255)		

Pruefung

Feldname	*Typ*	*Optionen*	*Bemerkung*
global_id	INTEGER	!1+P	Globale ID, wird von einer Funktion in einer Helper-Klasse erzeugt und ist für **alle anhängbaren Objekte eindeutig**
kategorie_id	INTEGER	F	Key der kategorie; Relation has wurde in Object hereingezogen, da es keine table object gibt wird dieses attribut weitervererbt (object, kategorie)
lva_id	INTEGER	F	zur Ausahl wird dem Benutzer allerdings die **LVA-NR** angezeigt
examiner	VARCHAR(255)		
grade	INTEGER		between 1 and 5

Notiz

Feldname	*Typ*	*Optionen*	*Bemerkung*
global_id	INTEGER	!1+P	Globale ID, wird von einer Funktion in einer Helper-Klasse erzeugt und ist für **alle anhängbaren Objekte eindeutig**
kategorie_id	INTEGER	F	Key der kategorie; Relation has wurde in Object hereingezogen, da es keine table object gibt wird dieses attribut weitervererbt (object, kategorie)
title	VARCHAR(255)		
note	VARCHAR(2000)		

Universitaet

Feldname	*Typ*	*Optionen*	*Bemerkung*
id	INTEGER	!1+P	
name	VARCHAR(255)		

Email

Feldname	*Typ*	*Optionen*	*Bemerkung*
id	INTEGER	!1+P	
kontakt_id	INTEGER	F	
email	VARCHAR(255)		

Kategorie

Feldname	*Typ*	*Optionen*	*Bemerkung*
id	INTEGER	!1+P	
title	VARCHAR(255)		
description	VARCHAR(255)		
shareable	BOOLEAN	!	

Termin_Kategorie

Feldname	*Typ*	*Optionen*	*Bemerkung*
id	INTEGER	!1+P	
name	VARCHAR(255)		

anhaengen (termincontainer, object)

Feldname	*Typ*	*Optionen*	*Bemerkung*
termincontainer_id	INTEGER	F[1]	
global_id	INTEGER	F[1]	Eindeutiger Schlüssel über alle anhängbaren Objekte
table_name	VARCHAR(255)		Datenbanktabelle, in der das angehängte Objekt liegt; prinzipiell nicht notwendig, allerdings besser für Performance, da dann nur diese Datenbanktabelle durchsucht werden muss

anhaengen (termin, object)

Feldname	*Typ*	*Optionen*	*Bemerkung*
termin_id	INTEGER	F[1]	
global_id	INTEGER	F[1]	Eindeutiger Schlüssel über alle anhängbaren Objekte
table_name	VARCHAR(255)		Datenbanktabelle, in der das angehängte Objekt liegt; prinzipiell nicht notwendig, allerdings besser für Performance, da dann nur diese Datenbanktabelle durchsucht werden muss