

Handbuch: ILIAS-Aufgabengenerator und Analysator

Digital Fellowship: ELEFDIA – Stand: 07/21

Bearbeitet von:

*Prof. Dr. Johanna May (Fellow),
Prof. Dr. Eberhard Waffenschmidt (Fellow)*

*Tobias Panteleit,
Maximilian Jakob Fröhlich,
Patrick Lehnen*

*Finanziert durch die gemeinsame Programmlinie Fellowship für Innovationen
in der digitalen Hochschullehre des Ministeriums für Kultur und Wissenschaft
des Landes Nordrhein-Westfalen und des Stifterverbandes*

Ministerium für
Kultur und Wissenschaft
des Landes Nordrhein-Westfalen



Inhaltsverzeichnis

TEIL A: ILIAS Aufgabenverwaltung und -erstellung	4
1 Wofür kann der Aufgabengenerator verwendet werden?	4
2 Installation und Start	5
2.1 Programm von Github beziehen	5
2.2 Setup Anaconda für ILIAS – Aufgabengenerator	7
2.2.1 Anaconda: Installationsanleitung für Windows 10	7
2.2.2 Spyder Programmierungsumgebung – Neues Projekt erstellen und Tool einbinden.....	10
3 Verwendung des Aufgabengenerators	12
3.1 Implementierungen für ILIAS und moodle	12
3.2 Erstellen von Fragen: Verwendung der GUI.....	13
3.3 Erstellen von Fragen: Verwendung einer Tabellenkalkulationsdatei	18
4 Beschreibung des Aufgabengenerators	19
4.1 Oberfläche GUI.....	19
4.1.1 Funktion: Wertebereiche berechnen	23
4.1.2 Funktion: Taxonomie verwalten	25
4.1.3 Funktion: Test-Einstellungen	27
4.1.4 Funktion: Excel-Datei importieren / Datenbank exportieren	29
4.2 Ordnerstruktur.....	30
4.2.1 Ablage generierter Fragen.....	30
4.2.2 Vorlagen für Tabellenkalkulationsdateien.....	30
4.2.3 Einbinden von Bildern über die Excel-Vorlage.....	31
4.3 XML-Struktur/Aufbau	32
4.3.1 Unterschied zwischen ILIAS-Test und ILIAS Pool	32
4.3.2 ILIAS Testeinstellungen (XML)	37
4.3.3 Aufbau der qpl.xml Dateien	47
4.3.4 Aufbau der qti.xml Dateien	48
4.3.4.1 Darstellung der XML Struktur für den Fragentyp „Formelfrage“	48
4.3.4.2 Darstellung der XML Struktur für den Fragentyp „Singlechoice“	52
4.3.4.3 Darstellung der XML Struktur für den Fragentyp „Multiplechoice“	55
4.3.4.4 Darstellung der XML Struktur für den Fragentyp „Zuordnungsfrage“	59
5 Alternative GUI zu Aufgabenverwaltung und -erstellung	62
5.1 Anwenderhinweise zur alternativen GUI.....	62
5.1.1 Erstmalige Verwendung der Software	62
5.1.2 Frage nachträglich bearbeiten.....	62
5.1.3 Test/ Fragenpool zusammenstellen.....	62

5.1.4	Hilfe	62
5.2	Software Aufbau der alternativen GUI	63
5.2.1	Kurze Beschreibung der .py Projektdateien	63
5.2.1.1	Disclaimer:.....	63
5.2.1.2	DB_Interface.py	63
5.2.1.3	Main.py	64
5.2.1.4	Test_Klasse.py.....	64
5.2.1.5	DB_Treeview.py	64
5.2.1.6	DB_creator.py	65
5.2.1.7	Variablen_Einfügen_UI.py.....	65
5.2.1.8	Picture_interface.py	65
5.2.1.9	XML_class.py.....	65
5.2.1.10	Time_input_UI.py	65
5.2.1.11	Fragen_GUI.py	66
5.2.2	Code Logik für die Integration neuer Funktionen.....	67
5.2.2.1	Index-Liste und Index-dict (Daten zuordnen)	67
6	Sonstiges	69
6.1	Ausführbare Datei aus Programmcode erstellen	69
6.1.1	Virtual-Environment aufsetzen	69
6.1.2	Notwendige Bibliotheken installieren und ausführbare Datei erstellen.....	70
TEIL B: Softwaredokumentation für ILIAS-Aufgabengenerator		72
1	Python Aufbau des Aufgabengenerators.....	72
1.1	Programmierung der GUI	72
1.2	Auflistung der Module und der zugehörigen Funktionen	72
1.2.1	Module für Fragentypen.....	74
1.2.1.1	Formelfrage.....	74
1.2.1.2	Formelfrage_Permutation.....	75
1.2.1.3	Singlechoice	76
1.2.1.4	Multiplechoice	76
1.2.1.5	Zuordnungsfrage.....	77
1.2.2	Modul zur Erstellung eines ILIAS Test/Pool	78
1.2.3	Modul für Testeinstellungen	78
1.2.4	Module für Datenbanken	79
1.2.4.1	Datenbanken anzeigen	79
1.2.4.2	Datenbanken erstellen und Import/Export (von/nach Tabellenkalkulationsdatei) 79	
1.2.4.3	Import bestehender ILIAS Test/Pool Dateien	79

1.2.5	Modul für Taxonomie/Textformatierung	80
1.2.6	Modul für Zeigerdiagramme	81
1.3	UML-Diagramme für Fragentyp-Module	82
1.3.1	Fragentyp: Formelfrage	82
1.3.2	Fragentyp: Singlechoice	83
1.3.3	Fragentyp: Multiplechoice	84
1.3.4	Fragentyp: Zuordnungsfrage	85
Teil C: Auswerte- und Analysetool		86
1 Nachkorrektur und Auswertung		86
1.4	Beschreibung des Nachkorrekturtools	86
1.5	Beschreibung des Auswertetools	91
1.5.1	Was ist Item Response Theory?	91
1.5.2	Nutzung des IRT-Tools	93
1.5.3	Erkenntnisse aus dem Prozess	95
2	Abbildungsverzeichnis	97
3	Tabellenverzeichnis	99

TEIL A: ILIAS Aufgabenverwaltung und -erstellung

1 Wofür kann der Aufgabengenerator verwendet werden?

Der Aufgabengenerator bietet die Möglichkeit, Fragen zügig und übersichtlich, wahlweise über eine GUI oder über eine Tabellenkalkulationsdatei, in eine Datenbank einzupflegen. Anschließend können die Fragen aus der Datenbank in einen Fragen-Test oder –Pool geschrieben werden. Die Datei wird dabei im, ILIAS konformen, XML Format geschrieben und kann nach der Erstellung direkt in Ilias importiert werden. Bei der Erstellung von Tests besteht die Möglichkeit Testeinstellungen vorzunehmen. Diese Einstellungen werden ebenfalls in einer Datenbank gespeichert und können über ein Profil direkt wiederverwendet werden. Ein ständiges erneutes auswählen von Standardeinstellungen bei der Erstellung von Tests, entfällt damit komplett.

Der große Vorteil bietet sich bei der Erstellung von Fragen über eine Tabellenkalkulationsdatei wie z.B. Excel. Als Beispiel soll hier der Fragentyp „Formelfrage“ dienen, bei der eine variable Formel eingegeben werden kann, die Ilias beim Start der Frage berechnet. Über die Excel-Tabelle können eigene Templates geschrieben werden, die sich auf die entsprechenden Formeln und Variablen beziehen, um diese auf Sinnhaftigkeit bzw. Fehler zu überprüfen. Erzeugt eine bestimmte Kombination der Variablen ein $\sqrt{-1}$, sollte dies nicht erst während des Tests erkannt werden.

Weiterhin bietet der Aufgabengenerator die Option Fragen zu permutieren. Fragen sollen in großer Zahl und automatisiert erstellt werden. Über einen Index können Text-Elemente, wie z.B. Variablen oder Formeln, verändert werden. Somit müssen ähnliche Fragen nicht mehrfach neu erstellt werden.

Der ILIAS-Aufgabengenerator steht als OER zur freien Verfügung.

2 Installation und Start

2.1 Programm von Github beziehen

Der ILIAS-Aufgabengenerator steht als OER zur Verfügung und kann über Github bezogen werden. Es besteht die Option das Programm via Anaconda zu starten oder die mitgelieferte Test_Generator.exe zu verwenden. Diese bietet den Vorteil das kein Setup notwendig ist und somit direkt verwendet werden kann.

Link: <https://github.com/TPanteleit/ILIAS---Test-Generator>

Der Link führt zur Github Übersichtsseite und sollte in etwa wie folgt aussehen:

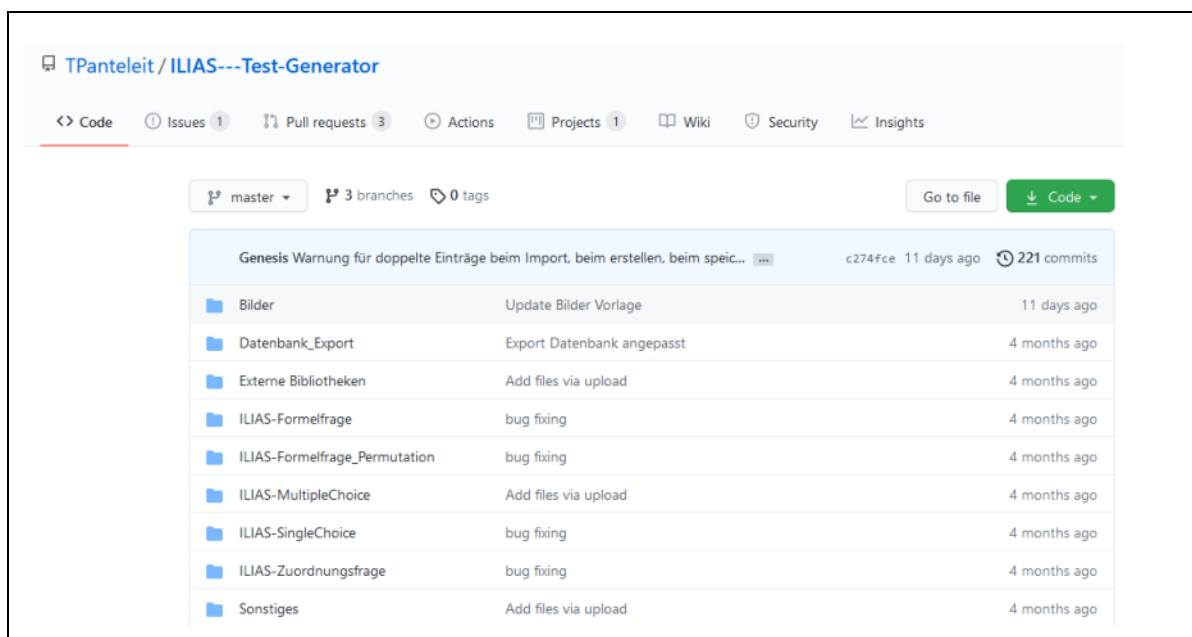


Abbildung 1: Darstellung des Programms auf Github

Über den grünen Button „Code“ kann das Programm heruntergeladen werden „Download ZIP“. Es kann immer nur das gesamte Verzeichnis heruntergeladen werden. Ein Download von einzelnen Ordnern ist über Github nicht möglich!

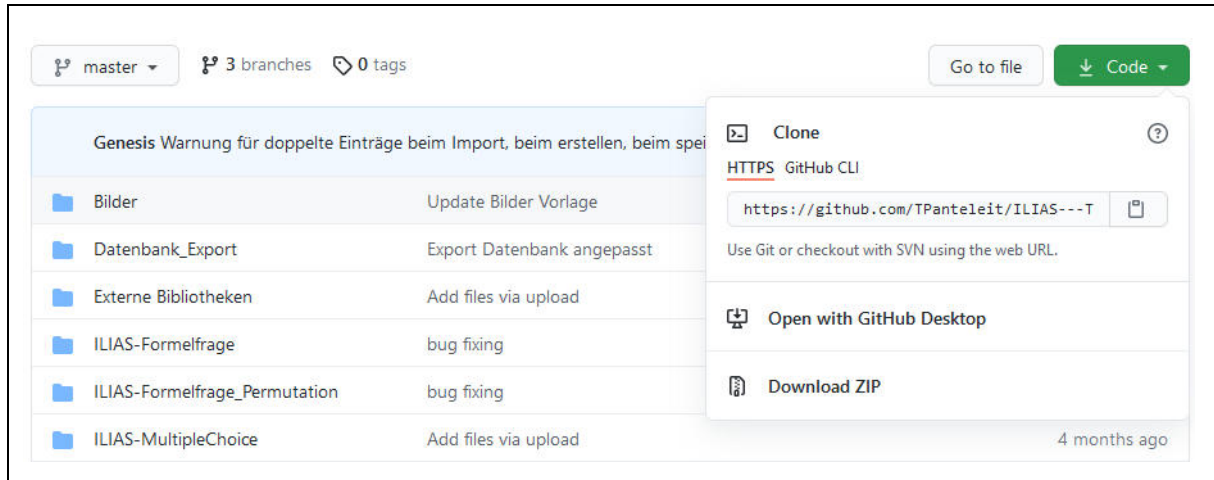


Abbildung 2: Programm von Github herunterladen

Wurde das Programm erfolgreich heruntergeladen, befindet sich im Zielordner eine gezippte Datei mit dem Namen „**ILIAS---Test-Generator-master**“. Im nächsten Schritt muss diese Datei entpackt werden. Der Inhalt des entpackten Ordners sollte dem folgenden entsprechen:

__Einstieg - ILIAS import	21.06.2021 13:26	Dateiordner	
Bilder	21.06.2021 13:26	Dateiordner	
Datenbank_Export	21.06.2021 13:26	Dateiordner	
Externe Bibliotheken	21.06.2021 13:26	Dateiordner	
ILIAS-Formelfrage	21.06.2021 13:26	Dateiordner	
ILIAS-Formelfrage_Permutation	21.06.2021 13:26	Dateiordner	
ILIAS-MultipleChoice	21.06.2021 13:26	Dateiordner	
ILIAS-SingleChoice	21.06.2021 13:26	Dateiordner	
ILIAS-Zuordnungsfrage	21.06.2021 13:26	Dateiordner	
Sonstiges	21.06.2021 13:26	Dateiordner	
Test_Generator_Datenbanken	21.06.2021 13:26	Dateiordner	
Test_Generator_Module	21.06.2021 13:26	Dateiordner	
Vorlage_für_Fragenpool	21.06.2021 13:26	Dateiordner	
.gitignore	21.06.2021 13:26	Textdokument	1 KB
Handbuch ILIAS-Testgenerator	21.06.2021 13:26	Adobe Acrobat D...	477 KB
ILIAS_Test_Generator	21.06.2021 13:26	JetBrains PyChar...	15 KB
ILIAS_Test_Generator_executable	21.06.2021 13:26	Anwendung	80.622 KB
ReadMe Änderungen v1.7.1 zu v2.0	21.06.2021 13:26	Microsoft Word-D...	17 KB
Vorlage (FF) - Import Excel-Datei in Date...	21.06.2021 13:26	Microsoft Excel-A...	12 KB
Vorlage (MC) - Import Excel-Datei in Dat...	21.06.2021 13:26	Microsoft Excel-A...	69 KB
Vorlage (MQ) - Import Excel-Datei in Dat...	21.06.2021 13:26	Microsoft Excel-A...	63 KB
Vorlage (SC) - Import Excel-Datei in Date...	21.06.2021 13:26	Microsoft Excel-A...	10 KB

Abbildung 3: Inhalt des entpackten ILIAS-Aufgabengenerators

2.2 Setup Anaconda für ILIAS – Aufgabengenerator

Anaconda ist eine Open-Source-Distribution für Windows, Linux und MacOS. Dieses Programm kann zur freien Verwendung herangezogen werden und stellt ein „all-in-one“ Paket zur Verwaltung von Python und den zugehörigen Bibliotheken dar.

2.2.1 Anaconda: Installationsanleitung für Windows 10

Anaconda kann unter folgendem Link für jedes Betriebssystem heruntergeladen werden:

Link: <https://www.anaconda.com/products/individual>

Im Folgenden wird das Setup für Anaconda kurz durchlaufen.

Wichtig

Bei der Installation von Anaconda ist darauf zu achten, dass im Installationsschritt „Erweiterte Optionen“ das Häkchen bei „PATH“ gesetzt ist um spätere Komplikationen zu vermeiden.

(Fehler! Verweisquelle konnte nicht gefunden werden.)



Abbildung 4: Anaconda Setup – Start (1/8)

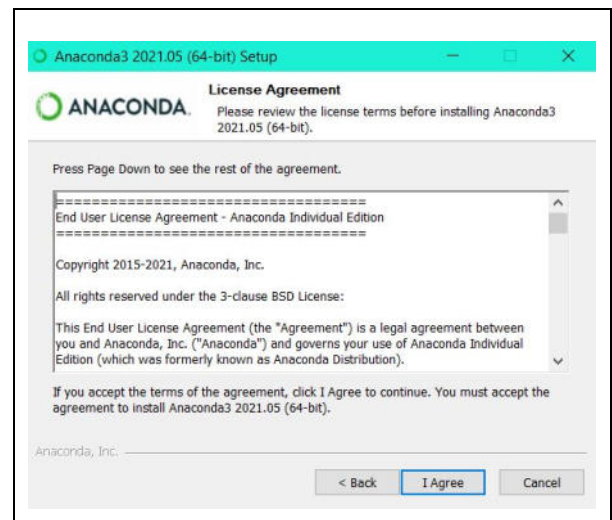


Abbildung 5: Anaconda Setup – Lizenzbedingungen (2/8)

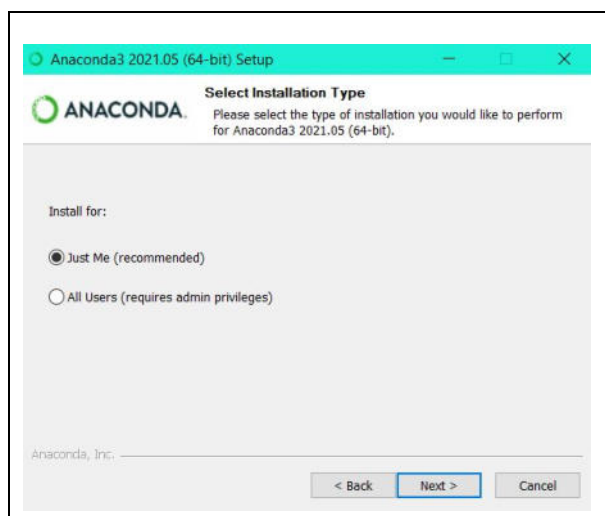


Abbildung 6: Anaconda Setup – Installationstyp (3/8)

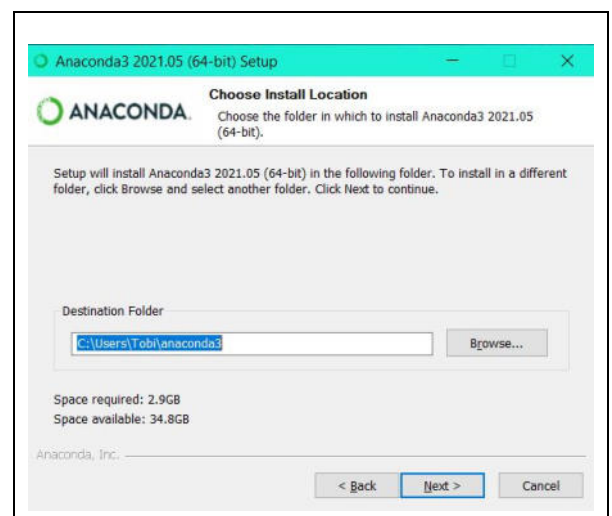


Abbildung 7: Anaconda Setup – Speicherort (4/8)

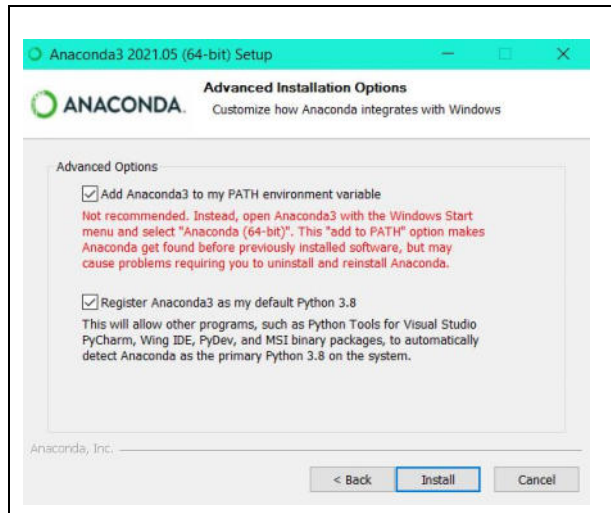


Abbildung 8: Anaconda Setup – Erweiterte Optionen (5/8)

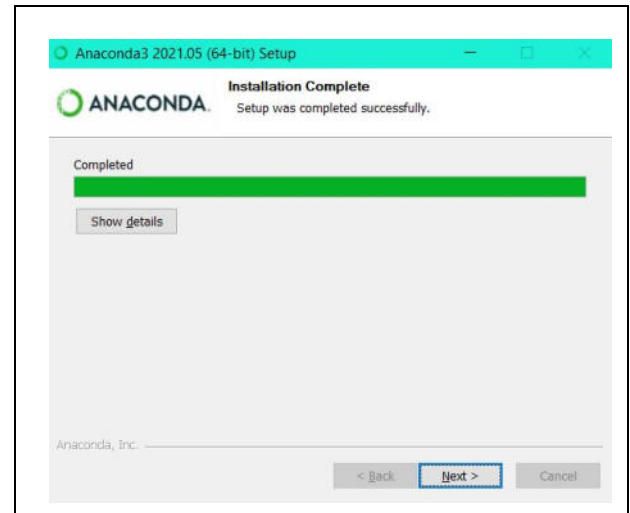


Abbildung 9: Anaconda Setup – Abgeschlossen (6/8)

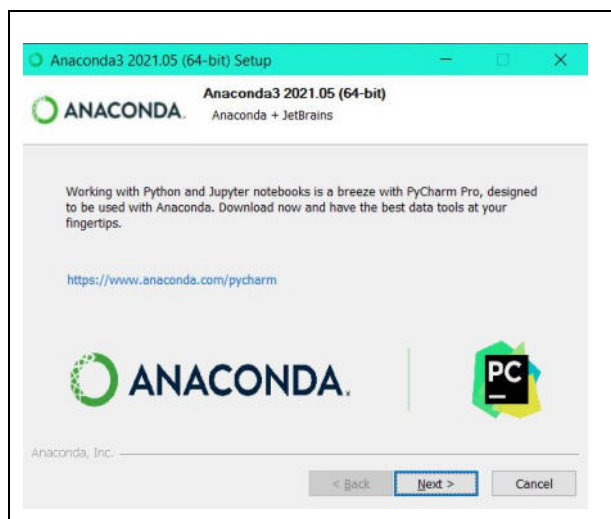


Abbildung 10: Anaconda Setup – Info JetBrains (7/8)

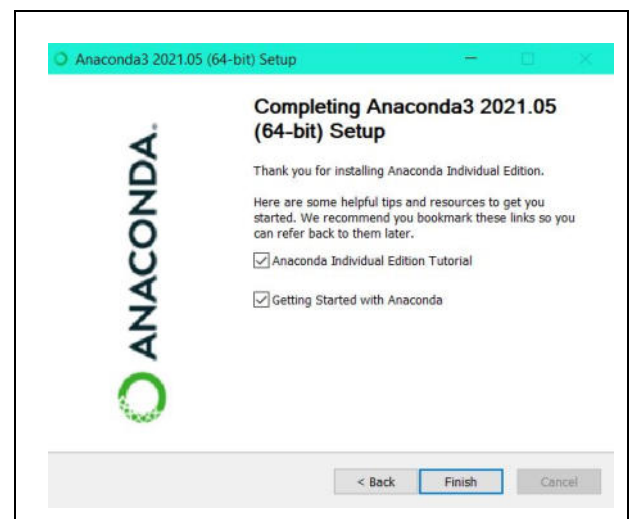


Abbildung 11: Anaconda Setup – Ende(8/8)

Um das Programm „ILIAS-Aufgabengenerator“ verwenden zu können, sind noch zusätzliche Bibliotheken notwendig, welche nicht „von Haus aus“ in Anaconda integriert sind. Diese Bibliotheken befinden sich im heruntergeladenen „ILIAS-Aufgabengenerator“-Ordner von Github, im Unterordner „Externe Bibliotheken“.

Es handelt sich um die folgenden beiden Pakete:

- tkScrolledFrame-1.0.4-py2.py3-none-any.whl
- Pmw-2.0.1.tar

Name	Änderungsdatum	Typ	Größe
 Pmw-2.0.1.tar	16.06.2020 19:39	WinRAR-Archiv	811 KB
 tkScrolledFrame-1.0.4-py2.py3-none-any.whl	27.05.2020 22:50	WHL-Datei	7 KB

Abbildung 12: Externe Bibliotheken

Die externen Bibliotheken sollten nicht entpackt werden!

Die beiden notwendigen Pakete müssen über das „Anaconda Prompt“ (Konsole) installiert werden. Dazu wird in der Windows Suchleiste „Anaconda Prompt“ eingegeben und, mit einem Rechtsklick, als **Administrator** gestartet.

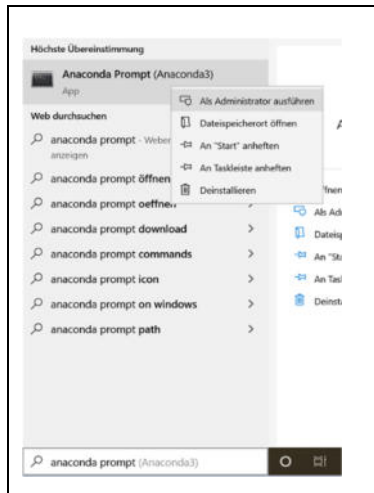


Abbildung 13: Öffnen der Konsole - Anaconda Prompt

In der Konsole ist nun der Pfad „(base) C:\Windows\system32“ zu sehen. Um die beiden externen Bibliotheken installieren zu können, muss zunächst das Verzeichnis zum heruntergeladenen „ILIAS – Aufgabengenerator“ gewechselt werden.

Mit dem Befehl „cd“ und einem anschließenden Pfad, wird das aktuelle Verzeichnis zum gewählten Pfad gesetzt. In diesem Beispiel liegt der heruntergeladene und entpackte Ordner auf dem Desktop. Wird der heruntergeladene Ordner von Github entpackt, befindet sich ein weiterer Ordner mit demselben Namen darin. Daher ist hier im Beispiel auch zweimal der Name „ILIAS---Test-Generator-master“ zu sehen.

Die Pakete werden nacheinander installiert, indem der Befehl „pip install“ gefolgt von Namen des Paketes (mit Endung!) in die Konsole eingegeben wird. Der Befehl kann aus der Bildbeschriftung kopiert werden.

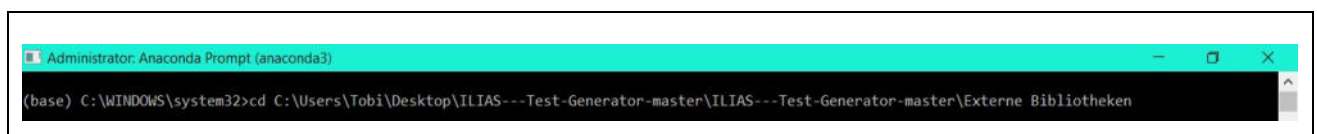


Abbildung 14: Anaconda Prompt - Verzeichnis wechseln



Abbildung 15: Anaconda Prompt – Installation Paket 1 -> pip install tkScrolledFrame-1.0.4-py2.py3-none-any.whl

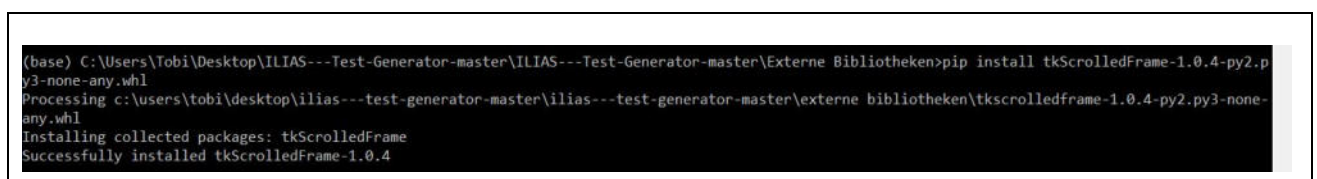


Abbildung 16: Anaconda Prompt – Installation Paket 2 -> pip install Pmw-2.0.1.tar.gz

Nach erfolgreicher Installation der einzelnen Pakete kann die Konsole geschlossen werden. Nun wird der Anaconda Navigator gestartet, der einige Programmierumgebungen zur Verfügung stellt. In diesem Beispiel wird „Spyder“ verwendet um ein neues Projekt anzulegen und den ILIAS-Aufgabengenerator zu verwenden.

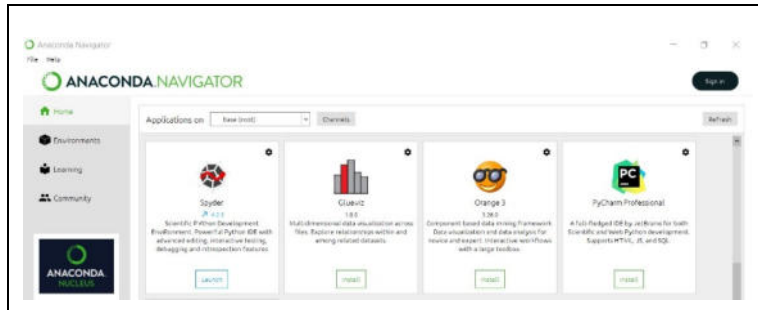


Abbildung 17: Übersicht Anaconda Navigator

2.2.2 Spyder Programmierumgebung – Neues Projekt erstellen und Tool einbinden

Um ein neues Projekt in Spyder zu erstellen, muss in der Funktionsleiste „Projects“ -> „New Project“ ausgewählt werden.

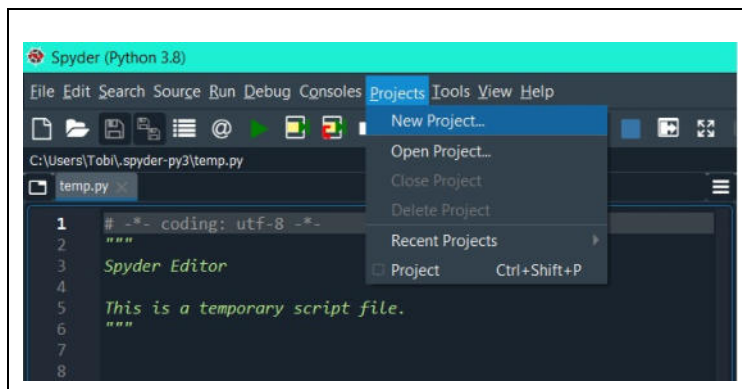


Abbildung 18: Spyder - Neues Projekt erstellen

Anschließend einen Projektnamen vergeben (hier „Neues Projekt“) und mit „Create“ erstellen

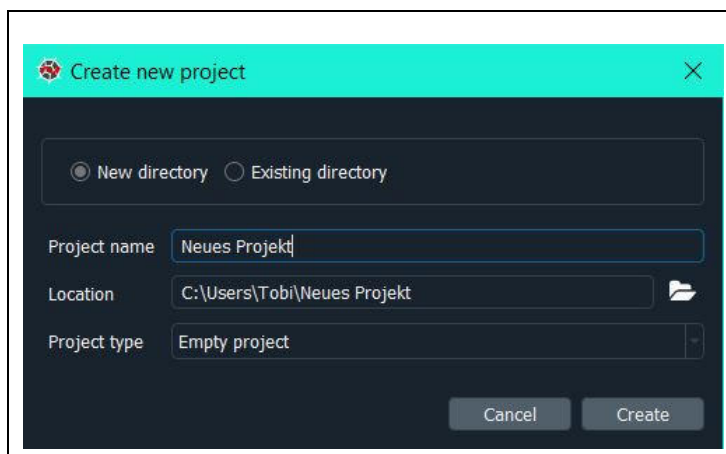


Abbildung 19: Spyder - Projektname & -verzeichnis

Nach erfolgreicher Erstellung kann in der linken Seite der Inhalt des Projekts eingesehen werden. Da dieses Projekt noch leer ist, sind keine weiteren Dateien sichtbar.

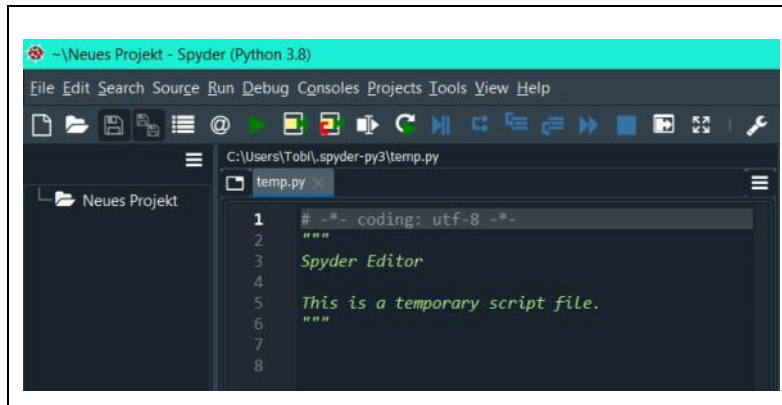


Abbildung 20: Spyder – Ansicht -> leeres Projekt

In das Projektverzeichnis soll nun der Inhalt aus dem „ILIAS-Aufgabengenerator“ übertragen werden. Dazu müssen die Dateien manuell in das Projektverzeichnis kopiert werden. Dabei gibt es keine Restriktionen und es kann der gesamte Inhalt aus dem „ILIAS-Aufgabengenerator“ übertragen werden.

Mit einem Rechtsklick auf die Datei „ILIAS_Test_Generator.py“ kann der „Run“ Befehl ausgewählt werden. Nach Betätigung wird die Oberfläche (GUI) des ILIAS-Aufgabengenerators gestartet und das Programm kann verwendet werden.

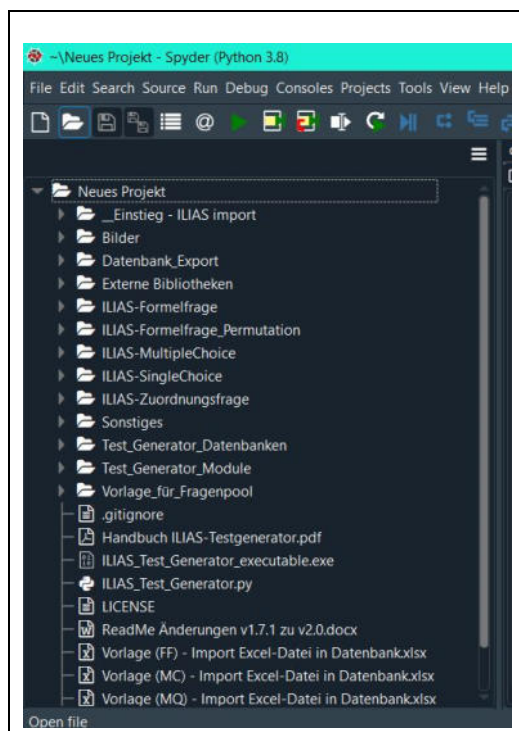


Abbildung 21: Spyder - Ansicht -> Vollständiges Projekt

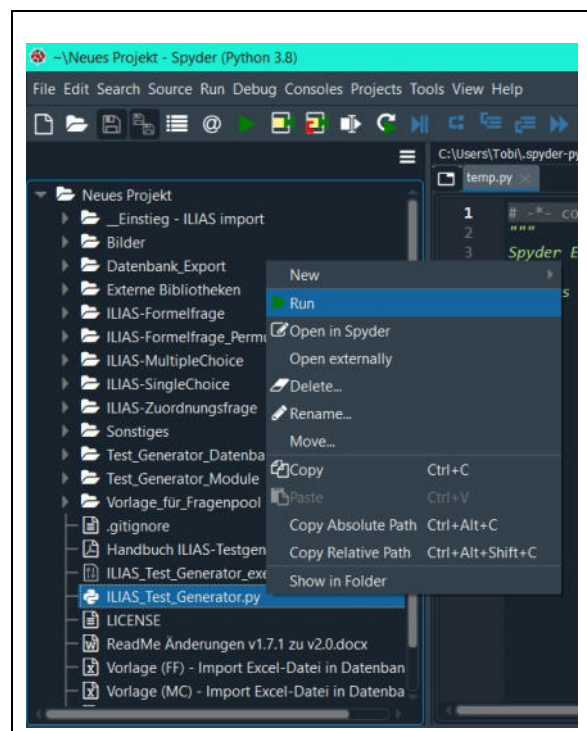


Abbildung 22: Spyder - ILIAS-Aufgabengenerator starten

3 Verwendung des Aufgabengenerators

3.1 Implementierungen für ILIAS und moodle

Ursprünglich war die Idee einen offline Aufgabengenerator zu entwickeln, der nach Möglichkeit die generierten Fragen einerseits für ILIAS als auch für moodle zur Verfügung stellen kann.

Diese Idee wurde später verworfen, da der Aufbau der Fragen (in der XML) für moodle deutliche Unterschiede zum ILIAS aufweist. Am Beispiel einer Formelfrage soll dieser Unterschied kurz angerissen werden.

Eine Formelfrage für ILIAS wird über eine Formel bestimmt. Diese Formel kann Variablen enthalten, die erst zur Laufzeit (von ILIAS) befüllt werden und diese werden definiert indem Minimum und Maximum angegeben werden.

Variable 1 - min: „0“, max. „10“

Variable 2 - min: „0“, max. „20“

Formel: $\$v1 + \$v2$

Zur Laufzeit werden für die Platzhalter „\$v1“ und „\$v2“ Werte von ILIAS bestimmt die innerhalb der definierten Grenzen liegen.

Bei jedem Aufruf der Frage ändern sich die Werte für die Variablen durch ILIAS.

Im Gegensatz hierzu wird eine Formelfrage für moodle bereits vorher ausgerechnet. Es wird eine Anzahl von „Sätzen“ bestimmt, die später zur Verfügung stehen sollen. Wird hier ebenfalls die Definierung der Variablen wie oben übernommen, kann anschließend ausgewählt werden, wie viele Sätze erstellt werden sollen (max. 100). Diese Sätze werden dann fest gespeichert und es wird nur noch innerhalb dieser Sätze variiert.

Z.B. Sätze = 3

1.) $0 + 14$

2.) $9 + 3$

3.) $5 + 10$

Es werden hier also lediglich drei Werte für Variable 1 und drei Werte für Variable 2 gespeichert. Bei einem Aufruf der Frage wird eine der drei Möglichkeiten angezeigt.

3.2 Erstellen von Fragen: Verwendung der GUI

Beim Start des Aufgabengenerators wird die GUI zur Erstellung einer Formelfrage dargestellt. Über die Reiter kann ein anderer Fragentyp ausgewählt werden. Anschließend können die fragenspezifischen Einträge vorgenommen werden.

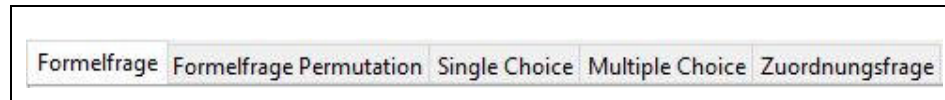


Abbildung 23: GUI - Auswahl von Fragentypen

Im Fragen-Text befindet sich der Inhalt der eigentlichen Frage. Hier können auch HTML spezifische Formatierungen vorgenommen werden (weitere Informationen hierzu im Kapitel 4.1).

Wichtig!

Für eine Formelfrage gilt:

Alle verwendeten Variablen und/oder Ergebnisse müssen im Fragen-Text auftauchen!

Fragen-Text

Gegeben ist ein Rechteck mit Breite $b = \$v1$ cm und Höhe $h = \$v2$ cm.
Berechne die Fläche $\$r1$ in cm.
Berechne die zweifache Fläche $\$r3$ in mm.

Bearbeitungsdauer

Std: 23 Min: 0 Sek: 0

Anzahl der Variablen: 3

	Min.	Max.	Präz.	Teilbar durch
Variable 1	0	10	1	1
Variable 2	0	20	1	1
Variable 3				

Anzahl der Ergebnisse: 5

	Min.	Max.	Präz.	Tol.	Punkte	Formel
Ergebnis 1	0.00	200.0	1	1	1	$\$v1 * \$v2$
Ergebnis 2	0.00	400.0	1	1	1	$\$r1 * 2$
Ergebnis 3	0.00	4000.0	1	1	1	$\$r2 * 10$

Abbildung 24: GUI (Formelfrage) - Hauptteil

Hier wird das Ergebnis „ $\$r2$ “ nicht im Fragen-Text verwendet. Ein Import in ILIAS wird zu einem Fehler führen. Ebenso verhält es sich mit definierten Variablen welche in der Formel auftauchen, aber nicht im Fragen-Text.

Die internen Namen der Variablen und Ergebnisse sind folgendermaßen deklariert.

- Variable 1 - $\$v1$, Variable 2 - $\$v2$ usw.
- Ergebnis 1 - $\$r1$, Ergebnis 2 - $\$r2$ usw.

Wird in der Formel „ $\$r1$ “ verwendet, wird die Definition von „Ergebnis 1“ angewandt.

Bei „ $\$r4$ “ die Definition von „Ergebnis 4“ usw.

Formelfrage:

Mit Hilfe der Auswahlboxen „Anzahl der Variablen“ und „Anzahl der Ergebnisse“, kann die Anzahl der Einträge variiert werden. Die leeren Felder unter den Auswahlboxen stehen für den Namen der Variable. Der Eintrag wird vom Programm/ILIAS nicht verwendet und dient lediglich der eigenen Orientierung.

Anzahl der Variablen:	3	Min.	Max.	Präz.	Teilbar durch
Variable 1	Breite	0	10	1	1
Variable 2	Höhe	0	20	1	1
Variable 3					

Abbildung 25: GUI (Formelfrage) - Deklaration von zwei Variablen

Wichtig!

Wird eine Variablen/Ergebnis-Zeile ausgefüllt, soll schließlich doch nicht verwendet werden, muss der Inhalt der Variablen-/Ergebnis -Zeile entfernt werden. Ein „ausblenden“ reicht hier nicht:

Als Beispiel werden hier zunächst drei Variablen definiert:

Anzahl der Variablen:	3	Min.	Max.	Präz.	Teilbar durch
Variable 1		0	10	1	1
Variable 2		0	20	1	1
Variable 3		0	30	1	1

Abbildung 26: GUI (Formelfrage) - Deklaration von drei Variablen

Anschließend wird sich dazu entschieden, dass drei Variablen zu viel sind und zwei Variablen ausreichen (Anzahl der Variablen von drei auf zwei geändert):

Anzahl der Variablen:	2	Min.	Max.	Präz.	Teilbar durch
Variable 1		0	10	1	1
Variable 2		0	20	1	1

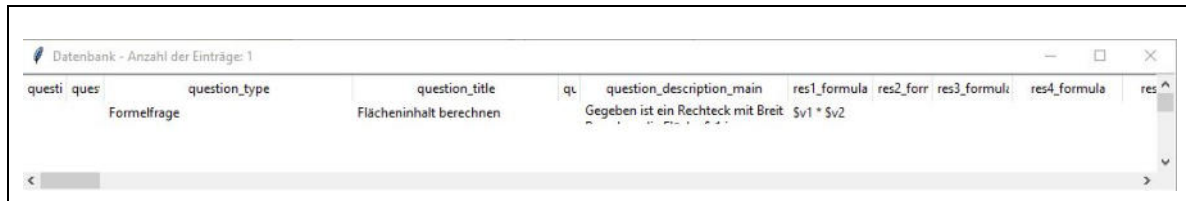
Abbildung 27: GUI (Formelfrage) - Ansicht von drei Variablen auf zwei geändert

Nun ist die Variable 3 nicht mehr sichtbar, aber dennoch definiert. Beim Auswählen von drei Variablen sind die zuvor eingetragenen Werte noch vorhanden. Hier müsste der Inhalt von Variable 3 entfernt werden:

Anzahl der Variablen:	3	Min.	Max.	Präz.	Teilbar durch
Variable 1		0	10	1	1
Variable 2		0	20	1	1
Variable 3					

Abbildung 28: GUI (Formelfrage) - Inhalt von Variable 3 wurde gelöscht

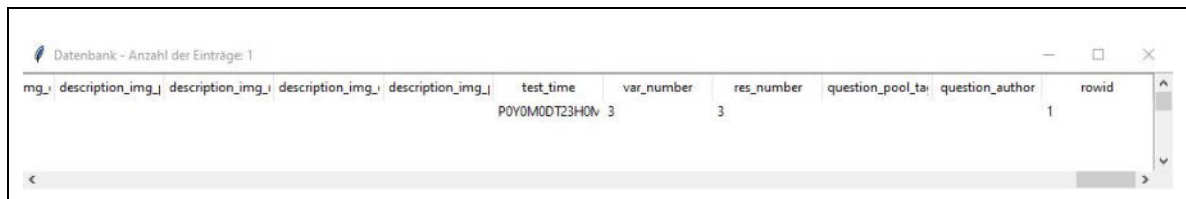
Wurden alle Einträge getätigt, wird mit einem Klick auf „Speichern unter neuer ID“ die Frage in der Datenbank gespeichert. Mit „FF – Datenbank anzeigen“ kann die Datenbank eingesehen werden.



question_type	question_title	question_description_main	res1_formula	res2_formula	res3_formula	res4_formula	res5_formula
Formelfrage	Flächeninhalt berechnen	Gegeben ist ein Rechteck mit Breit \$v1 * \$v2\$					

Abbildung 29: GUI (Datenbank) - Übersicht der Einträge (Beginn)

Wird die Ansicht nach rechts (bis zum Ende) verschoben, kann die ID (rowid) der Frage, innerhalb der DB, eingesehen werden:



mg_id	description_img_id	test_time	var_number	res_number	question_pool_tag	question_author	rowid
		POYOMODT3HON	3	3			1

Abbildung 30: GUI (Datenbank) - Übersicht der Einträge (Ende)

Um diese Frage nun in einen Test bzw. Pool schreiben zu können, ist die ID notwendig. Die Eingabefelder neben den Buttons „FF-Test erstellen“ bzw. „FF-Pool erstellen“ dienen der Eingabe der Fragen-IDs. In folgendem Beispiel wird die Frage mit der ID „1“ in einen Test geschrieben.

Abbildung 31: GUI (Formelfrage) – Erstellung eines Fragentests

Mit einem Klick auf „FF-Test erstellen“ wird der Test geschrieben. Auf der Konsole wird die erfolgreiche Erstellung angezeigt:

```
FORMELFRAGE: ILIAS-TEST WIRD ERSTELLT... ID: 1
1.) Formelfrage Frage erstellt! ---> Titel: Flächeninhalt berechnen
Überarbeite xml_datei_qti -- "&"-Zeichen... abgeschlossen!
```

Abbildung 32: Ausgabe in Konsole nach erstelltem Fragentest

Nun sollen mehrere Fragen in einen Pool geschrieben werden. Die IDs der Fragen müssen kommagetrennt in das Eingabefeld geschrieben werden.

Abbildung 33: GUI (Formelfrage) - Erstellung eines Fragenpools

Das Ergebnis wird, wie zuvor, in der Konsole dargestellt.

```
FORMELFRAGE: ILIAS-POOL WIRD ERSTELLT... ID: 1,3,4
1.) Formelfrage Frage erstellt! ---> Titel: Flächeninhalt berechnen
2.) Formelfrage Frage erstellt! ---> Titel: Würfel berechnen
3.) Formelfrage Frage erstellt! ---> Titel: Zylinder berechnen
Überarbeite xml_datei_qti -- "&"-Zeichen... abgeschlossen!
Taxonomie wird überarbeitet... abgeschlossen!
FRAGENPOOL ABGESCHLOSSEN
---> Erstellt im Ordner "ff_ilias_pool_abgabe\1596569820__0__qpl_1115823
```

Abbildung 34: Ausgabe in Konsole nach erstelltem Fragenpool

Über die Auswahlbox „Alle Einträge aus der DB erzeugen“ können direkt alle Fragen in einen Test oder Pool geschrieben werden. Es müssen keine IDs in das Eingabefeld geschrieben werden.

Der neu erstellte Test bzw. Pool kann anschließend im Projektordner gefunden werden. Es wird unterschieden nach Fragentyp und Test oder Pool. In diesem Beispiel wurde ein Fragen-Pool mit Formelfragen erstellt und ist demnach unter „Projektverzeichnis -> ILIAS-Formelfragen -> ff_ilias_pool_abgabe“ zu finden. Die Datei wird automatisch gezippt und kann direkt in ILIAS geladen werden „Fragenpool importieren“.

Verwendung von Test-Einstellungen:

Bei der Erstellung von Tests kann die Option zur Verwendung von Test-Einstellungen genutzt werden. Mit einem Klick auf „Test-Einstellungen“ öffnet sich ein neues Fenster in welcher die Einstellungen vorgenommen werden können. Wurden alle Einstellungen getätigt, muss ein neues Profil erstellt werden. Unter diesem Profil werden alle eingestellten Werte in einer Datenbank gespeichert. Im Eingabefeld „Speichern unter...“ wird ein Name eingetragen und mit „Speichern“ in der Datenbank angelegt.

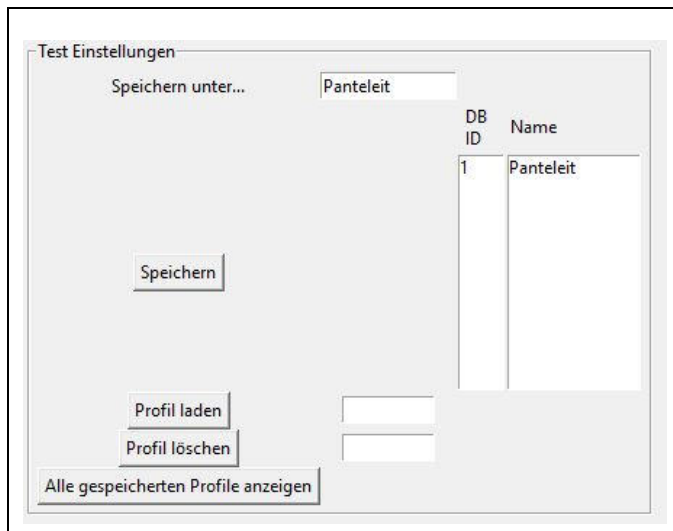


Abbildung 35: GUI (Testeinstellungen) - Übersicht der Profile

Sobald der Eintrag gespeichert wurde, kann die Ansicht wieder geschlossen werden. In der Übersicht kann nun unter „Test-Einstellungen verwenden?“ das angelegte Profil ausgewählt werden. Um die Ansicht der Auswahlbox zu aktualisieren, muss das Häkchen aktiviert werden.

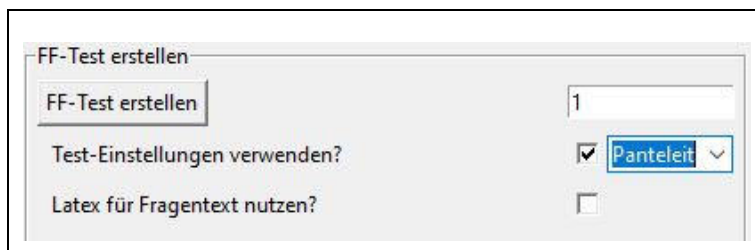


Abbildung 36: GUI (Formelfrage) - Test erstellen mit Testeinstellungen

3.3 Erstellen von Fragen: Verwendung einer Tabellenkalkulationsdatei

Fragen können, neben der Erstellung über die GUI, auch über eine Tabellenkalkulationsdatei erstellt werden. Hierzu kann die Vorlage „Vorlage (FF) - Import Excel-Datei in Datenbank“ verwendet werden. Für jeden Fragentypen existiert eine entsprechende Vorlage. Wird die Vorlage zur Ansicht geöffnet, kann folgendes Darstellung sichtbar (hier am Beispiel für Formelfrage):

	A	B	C	D	E	
1	question	question category	question type	question title	question description title	question description main
2	Typ A	Durchflutung	Formelfrage	7.11.01 Durchflutung berechnen	Berechnung der Durchflutung	Berechnen Sie die Durchflu
3	Typ A	Zylinderspule	Formelfrage	7.11.02 a Zylinderspule	Berechnung der magnetischen Feldstärke H in einer Spule	Eine luftgefüllte Zylinderspu
4	Typ A	Zylinderspule	Formelfrage	7.11.02 b Zylinderspule	Berechnung der magnetischen Flussdichte B in einer luftgefüllten Spule	Eine Zylinderspule wird mit
5	Typ A	Leiter einer Spule	Formelfrage	7.11.06 Kraft auf Leiter im Magnetfeld	Berechnung der Lorentzkraft auf einen Strom durchflossenen Leiter in ein	Eine Zylinderspule mit //n//
6	Typ A	Permeabilität von verschiedenen Materialien	Formelfrage	7.11.03 a Permeabilitätszahl von Eisen	magnetische Feldstärke ausrechnen	Betrachten Sie eine Kreisrin
7	Typ A	Primär- und Sekundärwerte ausrechnen	Formelfrage	8.9.03 a Idealer Transformator	Spannung auf der Primärseite berechnen	Ein idealer Transformator (U
8	Typ A	Primär- und Sekundärwerte ausrechnen	Formelfrage	8.9.03 b Idealer Transformator	Strom auf der Sekundärseite berechnen	Ein idealer Transformator (I
9	Typ A	Primär- und Sekundärwerte ausrechnen	Formelfrage	8.9.03 c Idealer Transformator	Strom auf der Primärseite berechnen	Ein idealer Transformator (I

Abbildung 37: Excel-Vorlage (Formelfrage) mit definierten Fragen

In der Zeile 1 sind die fragenspezifischen Einträge, mit gelbem Hintergrund, dargestellt. Jede Zeile unterhalb der „Zeile 1“ steht für jeweils eine Frage. Über die verschiedenen Spalten können die gleichen Einträge vorgenommen werden, wie über die GUI.

Ein Vorteil bei dieser Art der Fragerstellung besteht darin, dass Fragen schnell und übersichtlich erstellt werden können. Zusätzlich besteht die Möglichkeit, zum Beispiel für Formelfragen, eingetragene Formel über explizite Werte für die Variablen zu überprüfen. Dadurch werden Fehler in Formeln nicht erst zu Laufzeit erkannt.

4 Beschreibung des Aufgabengenerators

4.1 Oberfläche GUI

Bei einem Start des Programms erscheint die folgende grafische Oberfläche.

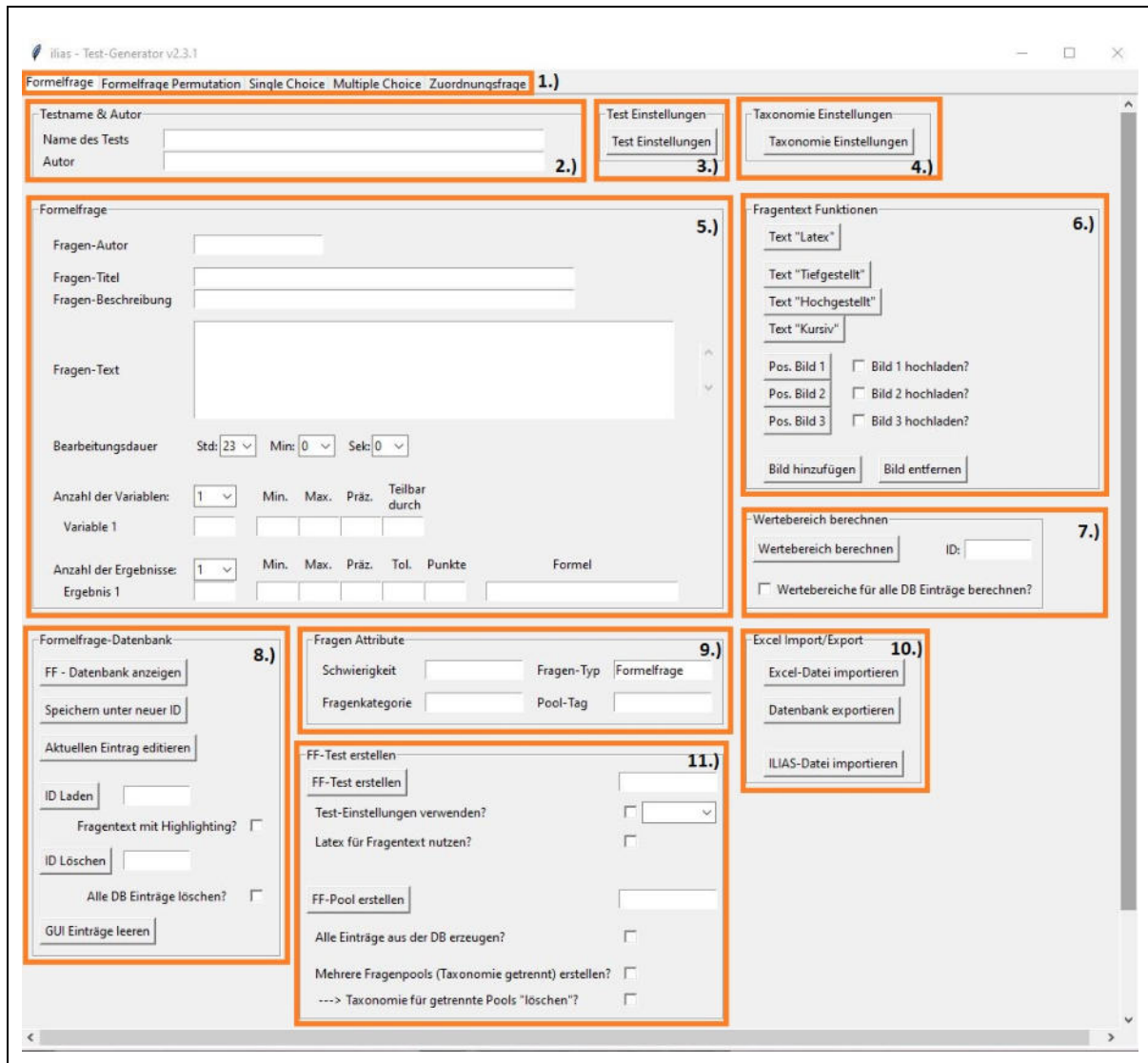


Abbildung 38: Übersicht GUI für Formelfrage

- 1.) Hierüber kann der Fragentyp ausgewählt werden, der bearbeitet bzw. verwaltet werden soll.
- 2.) Gibt den Testnamen und Autor bei Erstellung eines ILIAS-Aufgabentest oder –pool an.
- 3.) Hierüber können die Testeinstellungen für einen neuen Test vorgenommen werden
- 4.) Bietet die Möglichkeit die Taxonomie von bestehenden Fragenpools zu ändern
- 5.) Eingabemaske für die Formelfrage. Über „Anzahl der Variablen“ bzw. „Anzahl der Ergebnisse“ können zusätzliche Reihen angezeigt werden. Maximale Anzahl der Variablen ist derzeit auf „15“, und Ergebnisse auf „10“, beschränkt.

- 6.) Funktionen zur Bearbeitung des Fragentextes. Zur Verwendung von z.B. „Kursiv“, muss der Text in der Frage markiert und anschließend auf den Button „Kursiv“ gedrückt werden. Alternativ kann der Text auch manuell mit „//“ und „///“ umklammert werden-> //Beispiel/// „Tiefgestellt“ wird durch „_“ markiert. U_tief -> U_{tief}. „Hochgestellt“ wird durch „^“ markiert: I^2 -> I². „Latex“ wird durch „\ (“ bzw. „\ “ markiert: \ (Dieser Text ist in Latex formatiert \)



Abbildung 39: GUI - Layout von Fragen-Text bearbeiten (Beispiel)

Um verschiedene Symbole im Text darstellen zu können, besteht die Option HTML spezifische Kodierung zu verwenden (z.B. griechisches Alpha = Α bzw. α) .

Es besteht die Option Bilder zum Fragentext hinzuzufügen.

- Durch die Buttons „Pos. Bild 1/2/3“ wird der Text „%Bild1/2/3% “ in den Fragentext geschrieben. Bei der Erstellung eines Fragentests oder –pools wird an diese Stelle automatisch das entsprechende Bild eingefügt.
 - Durch Anklicken der Checkboxen „Bild 1/2/3 hochladen“ und anschließender Betätigung des „Bild hinzufügen“ Buttons, wird ein Auswahlfenster geöffnet, dass die Auswahl des Bildes ermöglicht.
 - **Die zu verwendeten Bilder müssen sich im Projektordner „Bilder“ befinden!**
- 7.) „Wertebereich berechnen“, ermöglicht die Berechnung der Formel die aktuell in der GUI eingetragen ist. Zur Berechnung werden die Variablengrenzen benötigt. Mit dem Eingabefeld „ID:“ kann eine explizite Frage aus der Datenbank berechnet werden. Die ID einer Frage kann in der Datenbank eingesehen werden (FF – Datenbank anzeigen, nach rechts bis zum Ende scrollen -> „rowid“).
- „Wertebereich für alle DB Einträge berechnen“ berechnet die Wertebereiche für alle Fragen aus der Datenbank. **Die vorher eingetragenen Werte werden dabei überschrieben!**

Anzahl der Variablen:		4	Min.	Max.	Präz.	Teilbar durch
Variable 1		1	10	1	1	
Variable 2		1	20	1	1	
Variable 3		1	30	1	1	
Variable 4						

Anzahl der Ergebnisse:		4	Min.	Max.	Präz.	Tol.	Punkte	Formel
Ergebnis 1		0.16	31.4159					(\$v1/\$v2)*pi
Ergebnis 2		0.16	942.477					\$r1*\$v3
Ergebnis 3		0.03	29608.8					\$r1*\$r2
Ergebnis 4								

Abbildung 40: Beispiel für Berechnung der Wertebereiche

8.) Bietet Funktionen zum Verwalten von Fragen in der Datenbank.

- **FF-Datenbank anzeigen:** Öffnet ein neues Fenster um Formelfragen in der Datenbank einsehen zu können.
- **Speichern unter neuer ID:** Speichert die aktuellen Einträge in der GUI in die DB.
- **Aktuellen Eintrag editieren:** Hierfür muss zunächst über „ID Laden“ ein Eintrag aus der Datenbank in die GUI geladen werden. Anschließend können die Einträge bearbeitet und durch „Eintrag editieren“ gespeichert werden. Es wird keine neue Frage erzeugt.
- **ID Laden:** Lädt eine Frage mit gewählter ID in die GUI.
„Fragentext mit Highlighting“ zeigt den Fragentext mit Farbformatierung an

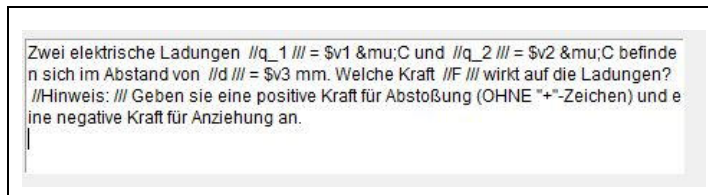


Abbildung 41: Fragentext ohne Highlighting laden

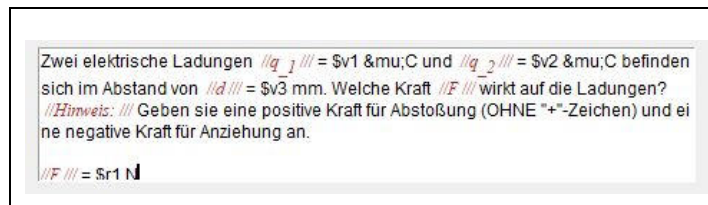


Abbildung 42: Fragentext mit Highlighting laden

- **ID Löschen:** Löscht eine Frage mit eingetragener ID
„Alle DB Einträge löschen“ löscht alle Einträge, mit Bestätigung via Popup.
Es gibt keine Backup Funktion!
- **GUI Einträge leeren:** Löscht die Einträge aller GUI Eingabefelder

9.) Hier werden Fragenattribute definiert. Mit Ausnahme von „Pool-Tag“ sind die Einträge für ILIAS nicht relevant. Pool-Tag beschreibt die Taxonomie. Hier können Knotennamen eingegeben werden, um die Frage einer bestimmten Taxonomie zuzuteilen.

10.) Excel Import und Export. Bietet Funktionen Fragen über eine Tabellenkalkulationsdatei einzulesen und die Datenbank als Tabellenkalkulationsdatei zu exportieren.

- **Excel-Datei einlesen:** Importiert Fragen aus einer Tabellenkalkulationsdatei „.xlsx“ oder „.ods“ in die Datenbank. Bereits bestehende Fragen (gleicher Fragentitel) werden nicht als neue Frage in die Datenbank importiert. Der bestehende Eintrag in der Datenbank wird editiert.
- **Datenbank exportieren:** Schreibt alle Fragen aus der Datenbank in eine Tabellenkalkulationsdatei „.xlsx“ oder „.ods“.
- **ILIAS-Datei importieren:** Liest bestehende Fragenpools (z.B. exportiert aus ILIAS) in die Datenbank ein.

11.) Ermöglicht die Erstellung von ILIAS-Fragentests oder –pools. In dem Eingabefeld neben den Buttons „FF-Test erstellen“ bzw. „FF-Pool erstellen“, werden die ID's der Fragen (Kommagetrennt) eingegeben. Anschließend wird durch die Betätigung des Buttons ein Test oder ein Pool generiert.

- **Test-Einstellungen verwenden:** Durch Auswahl eines Profils, können Test-Einstellungen übernommen werden.
- **Latex für Fragentext nutzen?** : Verwendet die Latex Konvertierung für den Fragentext.
- **Alle Einträge aus der DB erzeugen?:** Bei Betätigung und anschließender Verwendung von „FF-Test / Pool erstellen“ werden alle Fragen aus der Datenbank in einen Test bzw. Pool geschrieben.
- **Mehrere Fragenpools (Taxonomie getrennt) erstellen?:** Hierdurch wird nicht nur ein einziger Fragenpool mit allen Fragen verschiedener Taxonomien erstellt, sondern es werden mehrere Fragenpools mit jeweils eigener Taxonomie generiert.
„Taxonomie für getrennte Pools „löschen“ ?“: Die Fragenpools mit getrennter Taxonomie, besitzen hinterher keine Taxonomie-Zuordnung mehr.

4.1.1 Funktion: Wertebereiche berechnen

Für den Fragentyp „Formelfrage“ besteht die Option die eingegebenen Formeln mit den definierten Variablen zu berechnen. Dabei wird der „Min“ und „Max“ Eintrag vom entsprechenden Ergebnis Feld berechnet.

Dazu ist eine Schaltfläche auf der GUI (Formelfrage) vorhanden.

Bei der Benutzung von „Wertebereich berechnen“, ohne Eingabe einer ID, werden die aktuell in der GUI eingetragenen Formeln berechnet. Wird eine Fragen ID aus der Datenbank in das Feld eingetragen, werden die entsprechenden Werte dieser Frage berechnet und in der Datenbank gespeichert. Die alten eingespeicherten Werte werden dann unwiederbringlich überschrieben.

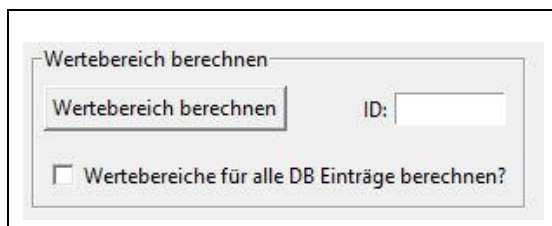


Abbildung 43: GUI (Formelfrage) - Wertebereich berechnen

Zusätzlich besteht die Option die Wertebereiche aller Formeln von sämtlichen Fragen aus der Datenbank zu berechnen. Wird diese Häkchen gesetzt und „Wertebereich berechnen“ gedrückt, folgt zunächst eine Warnung. Erst nach Bestätigung werden alle Formeln berechnet.

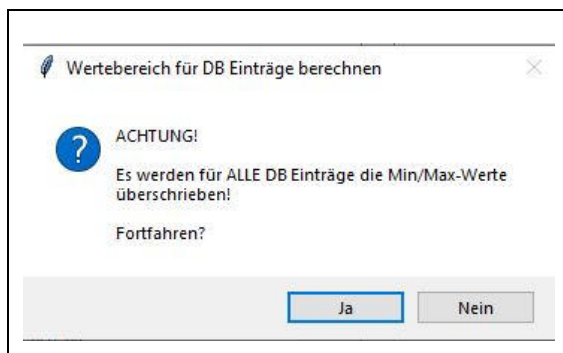


Abbildung 44: Wertebereich für alle DB-Einträge berechnen (Warnung)

Die Syntax der Formeln richtet sich dabei an die Vorgaben im ILIAS.

Auszug aus ILIAS für eine Formelfrage:

*Erlaubt ist die Verwendung von bereits definierten Variablen (\$v1 bis \$vn), von bereits definierten Ergebnissen (z.B. \$r1), das beliebige Klammern von Ausdrücken, die mathematischen Operatoren + (Addition), - (Subtraktion), * (Multiplikation), / (Division), ^ (Potenzieren), die Verwendung der Konstanten 'pi' für die Zahl Pi und 'e' für die Eulersche Zahl, sowie die mathematischen Funktionen 'sin', 'sinh', 'arcsin', 'asin', 'arcsinh', 'asinh', 'cos', 'cosh', 'arccos', 'acos', 'arccosh', 'acosh', 'tan', 'tanh', 'arctan', 'atan', 'arctanh', 'atanh', 'sqrt', 'abs', 'ln', 'log'.*

Im Folgenden soll eine Berechnung des Wertebereiches in einem Beispiel kurz angerissen werden.

Es werden zwei Variablen definiert und eine entsprechende Formel eingetragen. Durch einen Klick auf „Wertebereich berechnen“ werden die Werte für „Min“ und „Max“ für Ergebnis 1 direkt eingetragen. Zusätzlich wird das exakt berechnete Ergebnis in der Konsole angezeigt

Anzahl der Variablen:	2	Min.	Max.	Präz.	Teilbar durch		
Variable 1		1	10	1	1		
Variable 2		1	3	1	1		
Anzahl der Ergebnisse:	2	Min.	Max.	Präz.	Tol.	Punkte	Formel
Ergebnis 1		0.33	10.0				\$v1/\$v2
Ergebnis 2							

Abbildung 45: Wertebereich berechnen – Beispiel mit einer Formel

```

-----
---> $v1/$v2 --- NUMPY: row['a'] / row['b']
--- min: 0.3333333333333333 max: 10.0
-----

```

Abbildung 46: Wertebereich berechnen - Ausgabe auf der Konsole

Die Formeln können dabei beliebig kompliziert sein. Je länger die Formel jedoch wird, desto länger kann die Berechnung dauern.

Anzahl der Variablen:	4	Min.	Max.	Präz.	Teilbar durch		
Variable 1		1	10	1	1		
Variable 2		1	3	1	1		
Variable 3		1	5	1	1		
Variable 4		1	11	1	1		
Anzahl der Ergebnisse:	4	Min.	Max.	Präz.	Tol.	Punkte	Formel
Ergebnis 1		0.33	10.0				\$v1/\$v2
Ergebnis 2		1.00	1.73205				sqrt(\$v2)
Ergebnis 3		0.98	1.0				s((\$v4-\$v2)*pi/180) + 0*(\$v1+\$v3)
Ergebnis 4		3.60	539.264				(\$v1*10^-6)*(\$v2*10^-6)/(4*pi*8.8'

Abbildung 47: Wertebereich berechnen - Beispiel mit vier Formeln

Ebenfalls ist es möglich Formeln zu berechnen, die wiederum auf den Ergebnissen von zuvor berechneten Formeln beruhen

Anzahl der Variablen:	4	Min.	Max.	Präz.	Teilbar durch		
Variable 1		1	10	1	1		
Variable 2		1	20	1	1		
Variable 3		1	5	1	1		
Variable 4		1	11	1	1		
Anzahl der Ergebnisse:	4	Min.	Max.	Präz.	Tol.	Punkte	Formel
Ergebnis 1		2.00	30.0				\$v1+\$v2
Ergebnis 2		2.00	150.0				\$v3*\$r1
Ergebnis 3		4.00	60.0				2*\$r1
Ergebnis 4		6.00	210.0				\$r2+\$r3

Abbildung 48: Wertebereich berechnen - Beispiel mit zuvor errechneten Ergebnissen

4.1.2 Funktion: Taxonomie verwalten

Über den Knopf „**Taxonomie Einstellungen**“ kann die Taxonomie von bestehenden Fragenpools bearbeitet werden. Dazu muss ein Fragenpool Ordner ausgewählt werden.

Wichtig hierbei ist, dass der gewählte Fragenpool nicht gezippt ist.

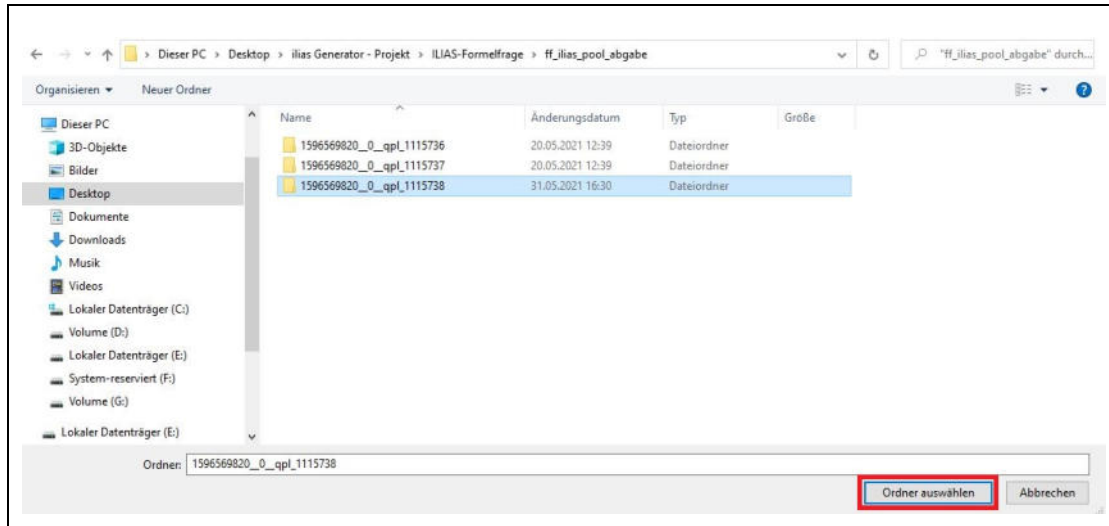


Abbildung 49: Auswahl des Fragenpools. Zielordner „anklicken“ und über „Ordner auswählen“ bestätigen

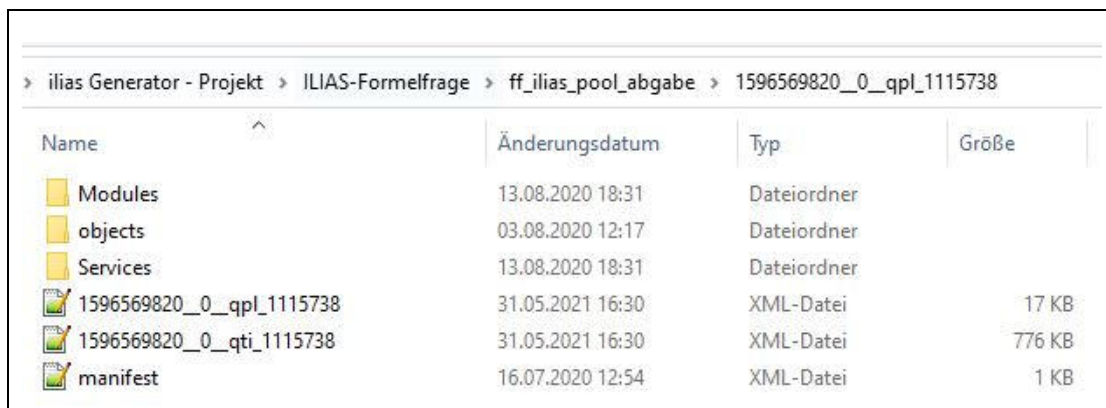


Abbildung 50: Inhalt des Zielordners

Nach erfolgreicher Auswahl öffnet sich eine Oberfläche zu Bearbeitung der Taxonomie des Fragenpools.

Fragen ID's

400000	----	01.1.01 Kraft auf Ladungen	---	1
400001	----	01.2.01 Elektrischer Strom	---	2
400002	----	01.5.1.01 Ladung, Strom, Spannung	---	2
400003	----	01.5.1.02 Arbeit um Ladung zu verschieben	---	2
400004	----	01.5.1.14 Strom und Ladung 1	---	2
400005	----	01.5.1.14 Strom und Ladung 2	---	2
400006	----	01.5.1.14 Strom und Ladung 3	---	2
400007	----	01.2.02 Elektrischer Strom	---	3
400008	----	03.1.01 Spannung	---	4
400009	----	05.2.10 Strom durch Widerstand	---	4
400010	----	05.2.11 Spannung aus Widerstand und Strom	---	4
400011	----	07.1.07 Widerstand Aufschrift	---	4
400012	----	03.2.01 Leitfähigkeit eines unbekannten Material	---	5
400013	----	03.2.02 Kupferdraht Querschnittsfläche	---	5
400014	----	03.2.03 Widerstand zwischen zwei Elektroden	---	5
400015	----	03.2.04 Widerstand eines Würfels	---	5
400016	----	03.2.05 Leitfähigkeit eines Würfels	---	5
400017	----	03.3.01 Länge eines runden Kupferdrahts	---	5
400018	----	03.3.03 Durchmesser eines runden Kupferdrahts	---	5
400019	----	03.3.04 Länge eines quadratischen Kupferdrahts	---	5

Fragen ID's

von Fragen ID bis Fragen ID

Taxonomie Baum

Name für Taxonomie

Name für Knoten

Vaterknoten

Taxonomie Bild

Taxonomie

- 1
- 10
- 11
- 12
- 2
- 3
- 4
- 5
- 6
- 7
- 8

Abbildung 51: Darstellung Fragenpool-Taxonomien

- In der linken Hälfte werden die enthaltenen Fragen des Fragenpools angezeigt. Die Ziffern am Anfang jeder Frage, stehen für die zugehörige Fragen-ID. Durch diese ID kann die Zugehörigkeit der Frage zu einer Taxonomie bearbeitet werden. Die derzeitige zugeordnete Taxonomie der Frage wird nach dem „---“ angezeigt. In diesem Fall Wurde die Frage „01.1.01 Kraft auf Ladungen“ der Taxonomie „1“ zugeordnet.
- In der rechten Hälfte befindet sich oben die Maske zum Ändern der Taxonomie von Fragen. Über die Fragen-ID's können Fragen ausgewählt werden und über die Drop-Down-Box wird die Taxonomie verändert

4.1.3 Funktion: Test-Einstellungen

Test-Einstellungen können zu jedem Fragen-Test vorgenommen werden. Die Einstellungen orientieren sich an den ILIAS-Einstellungen, wobei einige Werte jedoch nicht importiert werden können. Diese Einträge sind mit „not working“ deklariert.

In zukünftigen ILIAS Versionen besteht die Möglichkeit, dass weitere Einträge nicht importiert werden können!

Abbildung 52: GUI (Testeinstellungen) – 1/3

Abbildung 53: GUI (Testeinstellungen) – 2/3

DB ID	Name
1	Panteleit

Speichern unter:

Sämtliche Einträge können in einem Profil gespeichert werden. Hier muss ein Profil-Name angegeben werden.

Speichern:

Speichert sämtliche Einträge im definierten Profil-Namen

Profil laden:

Lädt die Einstellungen vom Profil. In die Eingabebox muss die ID des Profils eingetragen werden.

Profil löschen:

Löscht das Profil. In die Eingabebox muss die ID des Profils eingetragen werden.

Alle gespeicherten Profile anzeigen:

Hiermit werden die Einträge der Datenbank angezeigt.

Abbildung 54: GUI (Testeinstellungen) – 3/3

4.1.4 Funktion: Excel-Datei importieren / Datenbank exportieren

In die Datenbank des ILIAS-Aufgabengenerators können Fragen aus einer Tabellenkalkulationsdatei importiert werden. Dazu dient die Schaltfläche „Excel-Datei importieren“. Um einen reibungslosen Import zu gewährleisten, sollte die jeweilige Vorlage verwendet werden. Die Namen der Spalten in der Tabelle dürfen nicht verändert werden, die Reihenfolge spielt jedoch keine Rolle.

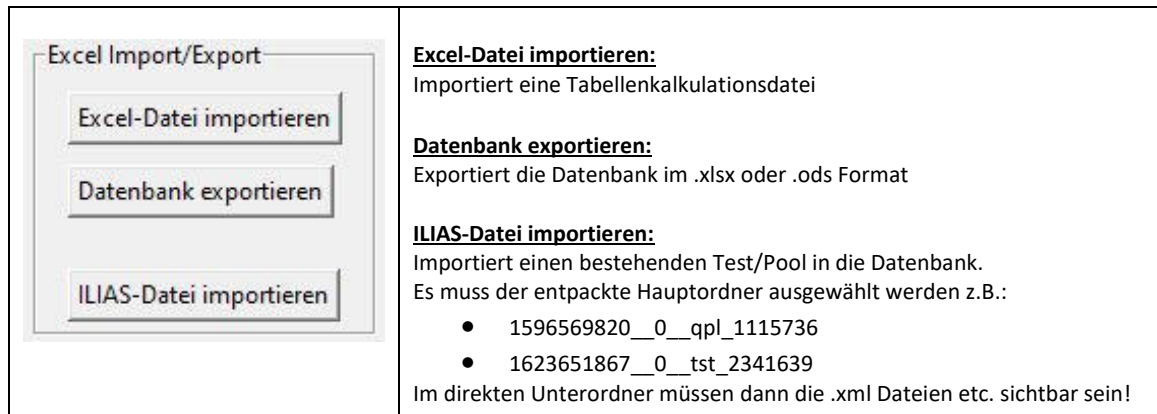


Abbildung 55: Fragen importieren

Bei einem Import wird die Anzahl der importierten Fragen in der Konsole angezeigt. Sollte in der Datenbank bereits eine Frage mit gleichem Titel vorhanden sein, wird die vorhandene Frage in der Datenbank editiert (ebenfalls in der Konsole sichtbar).

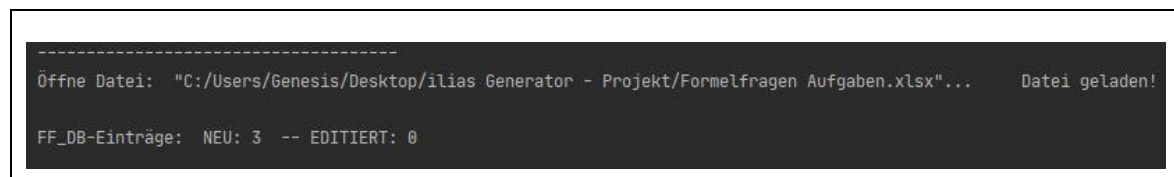


Abbildung 56: Import einer Excel-Datei - Fragen nicht in Datenbank vorhanden

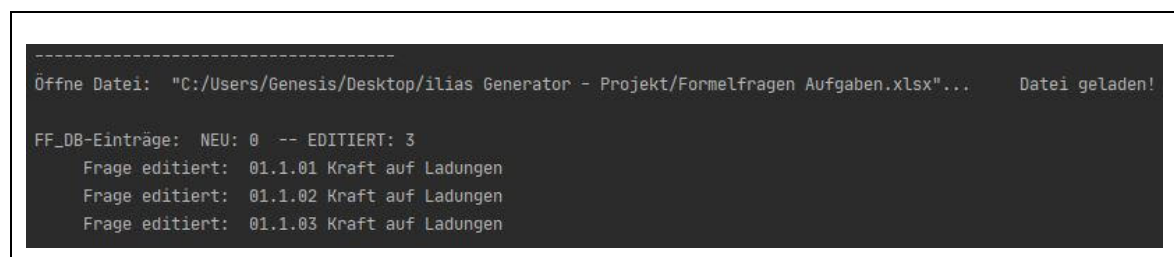


Abbildung 57: Import einer Excel-Datei – Fragen in Datenbank bereits vorhanden (werden editiert)

Sollten in der Tabellenkalkulationsdatei Fragen mit exakt gleichem Namen definiert worden sein, wird beim Import eine Warnung angezeigt. Die Fragen werden aber dennoch in die Datenbank importiert.

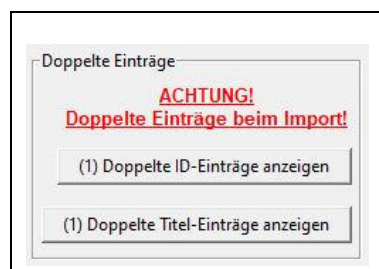


Abbildung 58: : Import einer Excel-Datei - Warnung bei doppelten Einträgen

4.2 Ordnerstruktur

4.2.1 Ablage generierter Fragen

Die Ablage von generierten Fragentests bzw. –pools ist unterteilt in die Fragentypen

- Formelfrage
- Formelfrage_Permutation
- Singlechoice
- Multiplechoice
- Zuordnungsfrage

Bei Erstellung von Fragentests bzw. –pools werden die Dateien, nach Fragentyp, sortiert abgelegt. Wird ein Fragenpool erstellt, befindet sich die Datei im Ordner:

- ***ILIAS-Formelfrage -> ff_ilias_pool_abgabe***
- ***ILIAS-SingleChoice -> sc_ilias_pool_abgabe***
- ***ILIAS-MultipleChoice -> mc_ilias_pool_abgabe***
- ***ILIAS-Zuordnungsfrage -> mq_ilias_pool_abgabe***

Die Ordner ID's werden jeweils inkrementiert und automatisch als ZIP konvertiert. Die Namen der Ordner dürfen nicht manuell umbenannt werden bzw. das Format muss beibehalten werden.

Kombination aus: 10-stelliger Nummer + „__0__qpl__“ + 7-stellige Nummer

Es ist nicht möglich eine ZIP Datei manuell nachträglich umzubenennen.

Die einfachste Möglichkeit, eine individuelle ID zu vergeben, ist einen Ordner mit der gewünschten ID in „.._ilias_pool_abgabe“ zu erstellen und anschließend über den Fragengenerator einen Fragenpool zu erstellen. Beim Erstellen werden die Namen der Ordner in „pool_abgabe“ ausgelesen. Die höchste vorkommende ID wird dann inkrementiert. Dabei wird sich nur auf die letzten 7 Ziffern beschränkt. Die vordere 10-stellige Nummer wird als FIX betrachtet.

4.2.2 Vorlagen für Tabellenkalkulationsdateien

Im Projektordner sind Excel-Datei Vorlagen enthalten, die für einen Fragenimport verwendet werden können. Die Dateien sind nach Fragentyp sortiert:

- Vorlage (FF) - Import Excel-Datei in Datenbank (Formelfrage)
- Vorlage (SC) - Import Excel-Datei in Datenbank (Singlechoice)
- Vorlage (MC) - Import Excel-Datei in Datenbank (Multiplechoice)
- Vorlage (MQ) - Import Excel-Datei in Datenbank (Zuordnungsfrage)

4.2.3 Einbinden von Bildern über die Excel-Vorlage

In der Vorlagedatei können Bilder über einen Link eingebunden werden. Der Link muss auf den „Bilder“ Ordner des Projektordners verweisen. Beispiel:

description_img_name_1	description_img_data_1	description_img_path_1
Bild_name	Placeholder	Bilder/Durchflutung berechnen.png

Tabelle 1: Tabellenkalkulationsdatei - Bilder einbinden

- Img_name_1: Hier muss Name für das Bild vergeben werden. Der Name kann frei gewählt werden
- Img_data_1: Wird ein Bild verwendet, muss hier „placeholder“ eingetragen werden. Bei einem Import in die Datenbank wird der placeholder gegen den Bild-Daten-String getauscht
- Img_path_1: Hier wird der Link zum Bild angegeben. Das Bild muss sich im Projektordner „Bilder“ befinden. **Die Endung des Bildes muss mit angegeben werden!**

4.3 XML-Struktur/Aufbau

4.3.1 Unterschied zwischen ILIAS-Test und ILIAS Pool

Ein ILIAS Test besteht im Grunde aus den gleichen Dateien wie ein ILIAS-Pool. Im Gegensatz zu einem ILIAS-Pool ist kein Eintrag für eine Taxonomie vorhanden, dafür besteht die Möglichkeit Test-Einstellungen vorzunehmen. Ein Fragenpool besteht lediglich aus der Auflistung von Fragen.

Wird die Dateistruktur zur Einsicht geöffnet, können folgende Dateien gefunden werden:

<u>ILIAS-Test</u>	<u>ILIAS-Pool</u>
<ul style="list-style-type: none"> • Modules -> Test -> set_1-> export.xml • objects -> il_0_mob_zzzzzzz -> *.png • Services -> Object -> set_1 -> export.xml • xxxxxxxxxxx_0__qti_yyyyyyy.xml • xxxxxxxxxxx_0__tst_yyyyyyy.xml • manifest.xml 	<ul style="list-style-type: none"> • Modules -> TestQuestionPool -> set_1-> export.xml • objects -> il_0_mob_xxxxxxx -> *.png • Services -> Taxonomy -> set_1 -> export.xml • xxxxxxxxxxx_0__qti_xxxxxxx.xml • xxxxxxxxxxx_0__qpl_xxxxxxx.xml • manifest.xml

Tabelle 2: Inhalt von ILIAS-Test und ILIAS-Pool

ILIAS-Test

... Test -> set_1 -> export.xml: Beinhaltet einen Eintrag der Fragen ID: „Id=yyyyyy“

objects -> il_0_mob_zzzzzzz: Beinhaltet die in den Fragen verwendeten Bilder. Die Nummerierung richtet sich nach der Fragen-ID aus der qti.xml.

... Object -> set_1 -> export.xml: Beinhaltet einen Eintrag der Fragen ID: „Id=yyyyyy“

Modules -> Test -> set_1-> export.xml (für Fragentest) Modules -> TestQuestionPool -> set_1-> export.xml (für Fragenpool)	
<pre> <exp:Export InstallationId="0" InstallationUrl="https://ilias.th-koeln.de" Entity="tst" SchemaVersion="4.1.0" TargetRelease="5.4.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:exp="http://www.ilias.de/Services/Export/exp/4_1" xsi:schemaLocation="http://www.ilias.de/Services/Export/exp/4_1 https://ilias.th-koeln.de/xml/ilias_export_4_1.xsd http://www.ilias.de/Modules/Test/htlm/4_1 https://ilias.th-koeln.de/xml/ilias_tst_4_1.xsd" xmlns="http://www.ilias.de/Modules/Test/htlm/4_1"> <exp:ExportItem Id="2040314"/> </exp:Export> </pre>	<p>In dieser XML-Datei müssen die Werte für Entity und exp:ExportItem angepasst werden. Entity bestimmt ob es sich um einen Fragentest („tst“) oder Fragenpool („qpl“) handelt.</p> <p>exp: ExportItem -> Id: Hier muss die aktuelle ID des Fragentests oder -pools stehen Bsp.: „1602067418_0__tst_30532“ -> Id=“30532“</p> <p>Die restlichen Einträge müssen nicht geändert werden bzw. sind „NameSpace“ Einträge. (Siehe Fehler! Verweisquelle konnte nicht gefunden werden.)</p>

Tabelle 3: Darstellung von XML-Datei: Modules -> Test -> set_1 -> export.xml

manifest.xml	
<pre> <Manifest MainEntity="tst" Title="Single choice Test" TargetRelease="5.4.0" InstallationId="0" InstallationUrl="https://ilias.th-koeln.de"> <ExportFile Component="Modules/Test" Path="Modules/Test/set_1/export.xml"/> <ExportFile Component="Services/Object" Path="Services/Object/set_1/export.xml"/> </Manifest> </pre>	<p>In dieser XML-Datei müssen die Werte für MainEntity und Title angepasst werden.</p> <p>MainEntity: Bestimmt ob es sich um einen Fragentest („tst“) oder Fragenpool („qpl“) handelt</p> <p>Title: Legt den Namen des Fragentests bzw. -pools fest</p> <p>Die restlichen Einträge müssen nicht geändert werden bzw. sind „NameSpace“ Einträge. (Siehe Fehler! Verweisquelle konnte nicht gefunden werden.)</p>

Tabelle 4: Darstellung von XML-Datei: manifest.xml

Die „export.xml“ im Unterordner „Services“ unterscheidet sich deutlich zwischen einem Fragentest und einem Fragenpool.

Für ein Fragenpool beinhaltet die „export.xml“ die Zuteilungen der Fragen zu den jeweiligen Taxonomien. Für einen Fragentest hingegen wird hier lediglich der ID-Nummer des Tests (z.B. „1602067418__0__tst_30532 -> 30532“)

Services -> Object -> set_1 -> export.xml (für Fragentest)	
<pre> <exp:Export InstallationId="0" InstallationUrl="https://ilias.th-koeln.de" Entity="common" SchemaVersion="5.4.0" TargetRelease="5.4.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:exp="http://www.ilias.de/Services/Export/exp/4_1" xsi:schemaLocation="http://www.ilias.de/Services/Export/exp/4_1 https://ilias.th-koeln.de/xml/ilias_export_4_1.xsd http://www.ilias.de/Services/Object/obj/5_4 https://ilias.th-koeln.de/xml/ilias_obj_5_4.xsd http://www.ilias.de/Services/DataSet/ds/4_3 https://ilias.th-koeln.de/xml/ilias_ds_4_3.xsd" xmlns=http://www.ilias.de/Services/Object/obj/5_4 xmlns:ds="http://www.ilias.de/Services/DataSet/ds/4_3"> <exp:ExportItem Id="2040314"> <ds:DataSet InstallationId="0" InstallationUrl="https://ilias.th-koeln.de" TopEntity="common"> <ds:Rec Entity="common"> <Common> <ObjId>2040314</ObjId> </Common> </ds:Rec> </ds:DataSet> </exp:ExportItem> </exp:Export> </pre>	<p>In dieser XML-Datei müssen die Werte für <code>exp:ExportItem</code> und <code><ObjId></code> angepasst werden.</p> <p>exp: ExportItem -> Id: Hier muss die aktuelle ID des Fragentests oder -pools stehen Bsp.: „1602067418__0__tst_30532“ -> Id=“30532“</p> <p>ObjId -> Text: Hier muss die aktuelle ID des Fragentests oder -pools stehen Bsp.: „1602067418__0__tst_30532“ -> Id=“30532“</p> <p>Die restlichen Einträge müssen nicht geändert werden bzw. sind „NameSpace“ Einträge. (Siehe Tabelle 3: Darstellung von XML-Datei: Modules -> Test -> set_1 -> export.xml)</p> <p>InstallationId, InstallationUrl, Entity, SchemaVersion, TargetRelease sind Einträge die nicht verändert werden müssen.</p> <p>xmlns:xsi, xmlns:exp, xsi:schemaLocation, xmlns, sind „NameSpace“ Einträge. Der Zusatz „exp:“ von „ExportItem“ steht für den „NameSpace“ von „xmlns:exp“. Das bedeutet, dass der ID Eintrag von ExportItem <u>nicht</u> durch „myroot.iter(exp:ExportItem)“ gefunden werden kann.</p> <p>Stattdessen muss der ganze NameSpace + „ExportItem“ angegeben werden: „myroot.iter('{ http://www.ilias.de/Services/Export/exp/4_1}ExportItem“):</p>

Tabelle 5: Darstellung von XML-Datei: Modules -> Test -> set_1 -> export.xml (für einen Fragentest)

Services -> Taxonomy -> set_1 -> export.xml (für Fragenpool)	
<pre> <?xml version="1.0" encoding="utf-8"?> <exp:Export InstallationId="0" InstallationUrl="https://ilias.th-koeln.de" Entity="tax" SchemaVersion="4.3.0" TargetRelease="5.4.0" Xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:exp="http://www.ilias.de/Services/Export/exp/4_1" xsi:schemaLocation="http://www.ilias.de/Services/Export/exp/4_1 https://ilias.th-koeln.de/xml/ilias_export_4_1.xsd http://www.ilias.de/Services/Taxonomy/tax/4_3 https://ilias.th-koeln.de/xml/ilias_tax_4_3.xsd http://www.ilias.de/Services/DataSet/ds/4_3 https://ilias.th-koeln.de/xml/ilias_ds_4_3.xsd" xmlns="http://www.ilias.de/Services/Taxonomy/tax/4_3" xmlns:ds="http://www.ilias.de/Services/DataSet/ds/4_3"> <exp:ExportItem Id="2423511"> <ds:DataSet InstallationId="0" InstallationUrl="https://ilias.th-koeln.de" TopEntity="tax"> <ds:Rec Entity="tax"> <Tax> <Id>2423511</Id> <Title>Taxonomie</Title> <Description/> <SortingMode>0</SortingMode> </Tax> </ds:Rec> <ds:Rec Entity="tax_tree"> <TaxTree> <TaxId>2423511</TaxId> <Child>43417</Child> <Parent>0</Parent> <Depth>1</Depth> <Type/> <Title>Root node for taxonomy 2423511</Title> <OrderNr>0</OrderNr> </TaxTree> </ds:Rec> <ds:Rec Entity="tax_tree"> <TaxTree> <TaxId>2423511</TaxId> <Child>43418</Child> <Parent>43417</Parent> <Depth>2</Depth> <Type>taxn</Type> <Title>Elektrotechnik</Title> <OrderNr>1</OrderNr> </TaxTree> </ds:Rec> </exp:ExportItem> </exp:Export> </pre>	<p>In dieser XML-Datei werden die Taxonomien festgelegt und die Fragen zugeordnet.</p> <p>In diesem Beispiel sind zwei untergeordnete Taxonomien definiert worden.</p> <ul style="list-style-type: none"> • „<ds:Rec Entity="tax">“ – steht dafür, dass eine Taxonomie Kategorisierung existiert. • „<Title>Taxonomie</Title>“ – Gibt den Hauptnamen der Taxonomie an. • „<Id>2423511</Id>“ – Muss identisch sein mit ID unter „<exp:ExportItem Id="2423511">“ <p>Im ILIAS entspricht dieser Eintrag folgendem Bild:</p> <div data-bbox="1086 518 1787 925"> </div> <p>Die beiden Einträge für „<ds:Rec Entity="tax_tree">“ geben weitere, untergeordnete, Taxonomien an. Die eigentlichen Bezeichnungen der Taxonomien sind hier nicht ersichtlich.</p> <ul style="list-style-type: none"> • <TaxId>2423511</TaxId> - Muss identisch sein mit ID unter „<exp:ExportItem Id="2423511">“ • <Child>43417</Child> - Gibt die ID dieser Taxonomie an • <Parent>0</Parent> - Gibt die ID der Taxonomie an. „0“ steht für die Haupttaxonomie. (hier „Elektrotechnik“) • <Depth>1</Depth> - Gibt an, dass diese Taxonomiestufe „1“ unter der Haupttaxonomie eingeordnet wird • <OrderNr>1</OrderNr> - Gibt die Reihenfolge an. Wird inkrementiert für jede Taxonomie.

Tabelle 6: Darstellung von XML-Datei: Modules -> TestQuestionPool -> set_1 -> export.xml (für einen Fragenpool) – 1/2

<pre> <ds:Rec Entity="tax_node_assignment"> <TaxNodeAssignment> <NodeId>43418</NodeId> <Component>qpl</Component> <ItemType>quest</ItemType> <ItemId>677964</ItemId> </TaxNodeAssignment> </ds:Rec> <ds:Rec Entity="tax_node_assignment"> <TaxNodeAssignment> <NodeId>43418</NodeId> <Component>qpl</Component> <ItemType>quest</ItemType> <ItemId>677965</ItemId> </TaxNodeAssignment> </ds:Rec> <ds:Rec Entity="tax_node_assignment"> <TaxNodeAssignment> <NodeId>43418</NodeId> <Component>qpl</Component> <ItemType>quest</ItemType> <ItemId>677966</ItemId> </TaxNodeAssignment> </ds:Rec> <ds:Rec Entity="tax_usage"> <TaxUsage> <TaxId>2423511</TaxId> <ObjId>2423470</ObjId> </TaxUsage> </ds:Rec> </ds:DataSet> </exp:ExportItem> </exp:Export> </pre>	<p>Im weiteren Verlauf der Datei werden dann die einzelnen Fragen den zuvor definierten "tax_tree" zugeordnet.</p> <p>In diesem Beispiel sind im Fragenpool drei Fragen definiert:</p> <ul style="list-style-type: none"> • <code><ds:Rec Entity="tax_node_assignment"></code> - steht für die Zuordnung einer Frage zu einer Taxonomie • <code><NodeId>43418</NodeId></code> - Gibt die Zuordnung der Frage zu "tax_tree" ID an In diesem Beispiel sind die Fragen der Taxonomie „Elektrotechnik“ zugeordnet • <code><Component>qpl</Component></code> - In jeder Frage gleich. (Fix Eintrag) • <code><ItemType>quest</ItemType></code> - In jeder Frage gleich. (Fix Eintrag) • <code><ItemId>677966</ItemId></code> - Gibt die ID der Frage an <p><code><ds:Rec Entity="tax_usage"></code></p> <ul style="list-style-type: none"> • <code><TaxId>2423511</TaxId></code> - Muss identisch sein mit „<code><exp:ExportItem Id="2423511"></code>“ • <code><ObjId>2423470</ObjId></code> - Muss identisch sein mit ID des Fragenpools Bsp.: Fragenpool - 1596569820__0__qpl_2423470 -> „2423470“
---	---

Tabelle 7: Darstellung von XML-Datei: Modules -> TestQuestionPool -> set_1 -> export.xml (für einen Fragenpool) – 2/2

4.3.2 ILIAS Testeinstellungen (XML)

<pre> <questestinterop> <assessment id="il_0_tst_11882" title="Zwischentest 09 Impedanz einer Induktivität"> <qticomment>alte Klausuraufgabe</qticomment> <duration>P0Y0M0D0T0H30M0S</duration> <qtimetadadata> <qtimetadadatafield> <fieldlabel>ILIAS_VERSION</fieldlabel> <fielentry>5.4.22 2021-05-14</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>anonymity</fieldlabel> <fielentry>0</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>use_pool</fieldlabel> <fielentry>0</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>question_set_type</fieldlabel> <fielentry>FIXED QUEST_SET</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>sequence_settings</fieldlabel> <fielentry>0</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>author</fieldlabel> <fielentry>Prof. Dr. Johanna May</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>reset_processing_time</fieldlabel> <fielentry>1</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>count_system</fieldlabel> <fielentry>0</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>mc_scoring</fieldlabel> <fielentry>0</fielentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>score_cutting</fieldlabel> <fielentry>0</fielentry> </qtimetadadatafield> </qtimetadadata> </assessment> </pre>	<p>assessment -> ident: Gibt den Namen des Tests an bestehend aus dem fix Eintrag „il_0_tst_“ mit anschließender variabler Nummer. Die Nummer darf maximal siebenstellig sein!</p> <p>assessment -> title: Bestimmt den Namen des Tests.</p> <p>qticomment: Hier steht die Fragentitel-Beschreibung.</p> <p>duration: Bestimmt die Dauer des Tests. Der Eintrag überschreibt <u>nicht</u> die Dauer der einzelnen Fragen!</p> <p>fieldlabel -> ILIAS_VERSION: Wird beim Exportieren aus dem ILIAS System geschrieben. Hat für den Import (bisher) keine Relevanz.</p> <p>fieldlabel -> anonymity: Datenschutz: „1“ - Testergebnisse ohne Namen / Anonymer Test „0“ - Testergebnisse mit Namen</p> <p>fieldlabel -> use_pool: Fragenpools verwenden: „1“ - Fragen aus Fragenpool hinzufügen oder direkt im Test erstellen. „0“ - Fragen nur im Test erstellen</p> <p>fieldlabel -> question_set_type: Auswahl der Testfragen: „FIXED QUEST_SET“ - Fest definierte Fragenauswahl. „RANDOM QUEST_SET“ - Zufällige Fragenauswahl. „DYNAMIC QUEST_SET“ - Wiedervorlagenmodus - alle Fragen eines Fragenpools.</p> <p>fieldlabel -> sequence_settings: Nicht beantwortete Fragen: „0“ – „Nicht beantwortete Fragen bleiben an ihrem Platz“ „1“ – „Nicht beantwortete Fragen werden ans Testende geschoben“</p> <p>fieldlabel -> author: Bestimmt den Author des Tests.</p> <p>fieldlabel -> reset_processing_time: Bearbeitungsdauer begrenzen: Maximale Bearbeitungsdauer für jeden Testlauf zurücksetzen: „0“ – Deaktiviert „1“ - Aktiviert</p> <p>fieldlabel -> count_system: Auswertung -> Bewertungssystem: „0“ – „Auch teilweise falsche Antworten erzielen Punkte“ „1“ – „Nur richtige und vollständige Antworten erzielen Punkte“</p> <p>fieldlabel -> mc_scoring: Auswertung -> Multiple-Choice-Fragen mit leerer Antwort: „0“ – Leere Antworten speichern Eine weitere Option für diesen Eintrag ist nicht vorhanden.</p> <p>fieldlabel -> score_cutting: Auswertung -> Negative Punkte: „0“ – „Negative Punkte pro Frage in ‚0 Punkte‘ ändern“ „1“ – „Negatives Gesamtergebnis in ‚0 Punkte‘ ändern“</p>
---	--

Tabelle 8: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 1/10

<pre> <qtimetadadatafield> <fieldlabel>password</fieldlabel> <fieldentry/> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>allowedUsers</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>allowedUsersTimeGap</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>pass_scoring</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>pass_deletion_allowed</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>reporting_date</fieldlabel> <fieldentry>P2021Y6M1DT15H50M0S</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>nr_of_tries</fieldlabel> <fieldentry>5</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>pass_waiting</fieldlabel> <fieldentry>00:000:00:00:00</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>kiosk</fieldlabel> <fieldentry>7</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>redirection_mode</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>redirection_url</fieldlabel> <fieldentry/> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>use_previous_answers</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> </pre>	<p>fieldlabel -> password: Legt ein Testpasswort fest.</p> <p>fieldlabel -> allowedUsers: Anzahl gleichzeitiger Teilnehmer begrenzen – Maximale Anzahl der Teilnehmer.</p> <p>fieldlabel -> allowedUsersTimeGap: Anzahl gleichzeitiger Teilnehmer begrenzen – Inaktivitätszeit der Teilnehmer.</p> <p>fieldlabel -> pass_scoring: Auswertung -> Bewertung bei mehreren Testdurchläufen: „0“ – „Letzten Testdurchlauf bewerten“ „1“ – „Besten Testdurchlauf bewerten“</p> <p>fieldlabel -> pass_deletion_allowed: Auswertung -> Durchläufe löschen: „0“ – „Die Löschung von vorherigen Durchläufen ist NICHT erlaubt“ „1“ – „Die Löschung von vorherigen Durchläufen ist erlaubt“</p> <p>fieldlabel -> reporting_date: Auswertung -> Teilnehmer sehen Testergebnisse: Zeitpunkt – „Ab definiertem Datum“ Eintrag für das Datum. Hier: „01.06.21 – 15:50 Uhr“ Ist nur sichtbar wenn die Einstellung „Zeitpunkt ab definiertem Datum“ aktiviert wurde.</p> <p>fieldlabel -> nr_of_tries: Anzahl von Testdurchläufen begrenzen – Maximale Anzahl von Testdurchläufen.</p> <p>fieldlabel -> pass_waiting: Wartezeit zwischen Durchläufen erzwingen. Formatierung: MM:DDD:HH:mm:ss (MM = Monat, DDD = Tag, HH = Stunden, mm = Minuten, ss = Sekunden) Die Eingabe von Sekunden ist in der ILIAS Maske nicht vorgesehen.</p> <p>fieldlabel -> kiosk: Prüfungsansicht: „0“ – Prüfungsansicht wird nicht verwendet. „1“ – Prüfungsansicht aktiviert, jedoch ohne „Titel des Tests“ und ohne „Name des Teilnehmers“ „3“ – Prüfungsansicht aktiviert, mit „Titel des Tests“ und ohne „Name des Teilnehmers“ „5“ – Prüfungsansicht aktiviert, ohne „Titel des Tests“ und mit „Name des Teilnehmers“ „7“ – Prüfungsansicht aktiviert, mit „Titel des Tests“ und mit „Name des Teilnehmers“</p> <p>fieldlabel -> redirection_mode: Weiterleitung: „0“ – Wird nicht verwendet. „1“ – Immer zur definierten Zielseite. „2“ – nur bei aktivierter Prüfungsansicht.</p> <p>fieldlabel -> redirection_url: Gibt die URL zur Website an die zur Weiterleitung verwendet werden soll.</p> <p>fieldlabel -> use_previous_answers: Verwendung vorheriger Lösungen</p>
---	---

Tabelle 9: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 2/10

```
<qtimetadafield>
  <fieldlabel>title_output</fieldlabel>
  <fieldentry>0</fieldentry>
</qtimetadafield>
<qtimetadafield>
  <fieldlabel>results_presentation</fieldlabel>
  <fieldentry>3</fieldentry>
</qtimetadafield>
<qtimetadafield>
  <fieldlabel>examid_in_test_pass</fieldlabel>
  <fieldentry>1</fieldentry>
</qtimetadafield>
<qtimetadafield>
  <fieldlabel>examid_in_test_res</fieldlabel>
  <fieldentry>0</fieldentry>
</qtimetadafield>
<qtimetadafield>
  <fieldlabel>show_summary</fieldlabel>
  <fieldentry>13</fieldentry>
</qtimetadafield>
```

fieldlabel -> title_output:

Anzeige der Fragentitel: „0“ – „Fragentitel und erreichbare Punkte“

„1“ – „Nur Fragentitel“

„2“ – „Weder Fragentitel noch erreichbare Punkte“

fieldlabel -> results_presentation:

Je nach Einstellungen, in der Kategorie „Auswertung“ -> „Details Testergebnisse“, wird hier eine ID angezeigt. Da es sehr viele Kombinationen gibt, wird hier nur ein Auszug gezeigt:

Bewertete Teilnehmerantworten	Rückmeldungen	Inhalte zur Wiederholung	Druckbare Liste der Antworten	Zeige bestmögliche Lösung	Druckausgabe der Ergebnisse	Id
0	0	0	0	0	0	0
1	0	0	0	0	0	2
1	1	0	0	0	0	10
1	1	1	1	0	0	78
1	1	1	1	1	0	206
1	1	1	1	1	1	222

fieldlabel -> examid_in_test_pass

ILIAS-Prüfungsnummer anzeigen: „0“ – Deaktiviert

„1“ - Aktiviert

fieldlabel -> examid_in_test_res:

Auswertung -> ILIAS-Prüfungsnummer anzeigen: „0“ – Deaktiviert

„1“ - Aktiviert

fieldlabel -> show_summary:

Fragenliste und Bearbeitungsstand anzeigen:

„0“ – Deaktiviert

„1“ – Aktiviert, ohne „Noch vor der ersten Frage anzeigen“,
ohne „Vor dem Ende des Tests anzeigen“,
ohne „Fragenbeschreibungen anzeigen“

„3“ – Aktiviert, mit „Noch vor der ersten Frage anzeigen“,
ohne „Vor dem Ende des Tests anzeigen“,
ohne „Fragenbeschreibungen anzeigen“

„5“ – Aktiviert, ohne „Noch vor der ersten Frage anzeigen“,
mit „Vor dem Ende des Tests anzeigen“,
ohne „Fragenbeschreibungen anzeigen“

„7“ – Aktiviert, mit „Noch vor der ersten Frage anzeigen“,
mit „Vor dem Ende des Tests anzeigen“,
ohne „Fragenbeschreibungen anzeigen“

„9“ – Aktiviert, ohne „Noch vor der ersten Frage anzeigen“,
ohne „Vor dem Ende des Tests anzeigen“,
mit „Fragenbeschreibungen anzeigen“

„11“ – Aktiviert, mit „Noch vor der ersten Frage anzeigen“,
ohne „Vor dem Ende des Tests anzeigen“,
mit „Fragenbeschreibungen anzeigen“

„13“ – Aktiviert, ohne „Noch vor der ersten Frage anzeigen“,
mit „Vor dem Ende des Tests anzeigen“,
mit „Fragenbeschreibungen anzeigen“

„15“ – Aktiviert, mit „Noch vor der ersten Frage anzeigen“,
mit „Vor dem Ende des Tests anzeigen“,
mit „Fragenbeschreibungen anzeigen“

Tabelle 10: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 3/10

<pre> <qtimetadafield> <fieldlabel>score_reporting</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>solution_details</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>print_bs_with_res</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>instant_verification</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>answer_feedback</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>answer_feedback_points</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> </pre>	<p>fieldlabel -> score_reporting: Auswertung -> Teilnehmer sehen Testergebnisse: „0“ – Deaktiviert „2“ – Aktiviert, „Zeitpunkt: Sofort“ „1“ – Aktiviert, „Zeitpunkt: Nach Testdurchlauf“ „4“ – Aktiviert, „Zeitpunkt: Nach Bestehen des Tests“ „3“ – Aktiviert, „Zeitpunkt: Ab definiertem Zeitpunkt“</p> <p>fieldlabel -> solution_details: Auswertung -> Bewertete Teilnehmerantworten: „0“ – Deaktiviert „1“ - Aktiviert</p> <p>fieldlabel -> print_bs_with_res: Auswertung -> Bewertete Teilnehmerantworten: Bestmögliche Antwort: „0“ – Deaktiviert „1“ - Aktiviert</p> <p>fieldlabel -> instant_verification: Direkte Rückmeldung: Inhalte der Rückmeldung: „0“ – Deaktiviert „1“ – „Bestmögliche Lösung anzeigen“ aktiviert</p> <p>fieldlabel -> answer_feedback: Direkte Rückmeldung: Inhalte der Rückmeldung: „0“ – Deaktiviert „1“ – „Rückmeldung zur richtigen Lösung“ aktiviert</p> <p>fieldlabel -> answer_feedback_points: Direkte Rückmeldung: Inhalte der Rückmeldung: „0“ – Deaktiviert „1“ – „Erreichte Punkte“ aktiviert</p>
---	--

Tabelle 11: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 4/10

<pre> <qtimetadafield> <fieldlabel>follow_qst_answer_fixation</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>instant_feedback_answer_fixation</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>force_instant_feedback</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_enabled</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_anon</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_achieved_ts</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_score</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_percentage</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_hints</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_wtime</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>highscore_own_table</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> </pre>	<p>fieldlabel -> follow_qst_answer_fixation: Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> instant_feedback_answer_fixation: Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> force_instant_feedback: Direkte Rückmeldung: Auslöser der Rückmeldung: „0“ – „Teilnehmer können die Rückmeldung selbst auslösen“ „1“ – „Die Rückmeldung wird bei Beantwortung von Fragen gezeigt“</p> <p>fieldlabel -> highscore_enabled: Auswertung -> Platzierung: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> highscore_anon: Auswertung -> Platzierung: Keine Namen anzeigen – „0“ Deaktiviert „1“ Aktiviert</p> <p>fieldlabel -> highscore_achieved_ts: Auswertung -> Platzierung: Datum – „0“ Deaktiviert „1“ Aktiviert</p> <p>fieldlabel -> highscore_score: Auswertung -> Platzierung: Punkte – „0“ Deaktiviert „1“ Aktiviert</p> <p>fieldlabel -> highscore_percentage: Auswertung -> Platzierung: Prozentwert – „0“ Deaktiviert „1“ Aktiviert</p> <p>fieldlabel -> highscore_hints: Auswertung -> Platzierung: Lösungshinweise – „0“ Deaktiviert „1“ Aktiviert</p> <p>fieldlabel -> highscore_wtime: Auswertung -> Platzierung: Bearbeitungsdauer – „0“ Deaktiviert „1“ Aktiviert</p> <p>fieldlabel -> highscore_own_table: Auswertung -> Platzierung: Modus – „Eigenen Rang anzeigen“: „0“ – Deaktiviert „1“ – Aktiviert</p>
--	---

Tabelle 12: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 5/10

<pre> <qtimetadadatafield> <fieldlabel>highscore_top_table</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>highscore_top_num</fieldlabel> <fieldentry>10</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>show_cancel</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>show_marker</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>fixed_participants</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>showfinalstatement</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>showinfo</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>mailnotification</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>mailnotttype</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>exportsettings</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>forcejs</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> </pre>	<p>fieldlabel -> highscore_top_table: Auswertung -> Platzierung: Modus – „Bestenliste“: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> highscore_top_num: Auswertung -> Platzierung: Länge der Bestenliste. Hier „10“ Platzierungen.</p> <p>fieldlabel -> show_cancel „Test unterbrechen“ anzeigen: „0“ – Nicht aktiv „1“ – Aktiviert</p> <p>fieldlabel -> show_marker Fragen markieren: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> fixed_participants Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> showfinalstatement Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> showinfo Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> mailnotification Benachrichtigung: Inhalt der Benachrichtigung: „0“ – Deaktiviert Inhalt der Benachrichtigung: „1“ – „Nur Benutzername und Datum des Beendens“ Inhalt der Benachrichtigung: „2“ – „Komplettes Testergebnis pro Benutzer“</p> <p>fieldlabel -> mailnotttype Benachrichtigung: Inhalt der Benachrichtigung: „0“ – Deaktiviert Inhalt der Benachrichtigung: „1“ – „Ein Versand findet auch beim Beenden einzelner Testdurchläufe statt“</p> <p>fieldlabel -> exportsettings Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> forcejs Eintrag konnte im ILIAS nicht zugeordnet werden</p>
--	---

Tabelle 13: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 6/10

<pre> <qtimetadadatafield> <fieldlabel>customstyle</fieldlabel> <fieldentry/> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>shuffle_questions</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>processing_time</fieldlabel> <fieldentry>00:30:00</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>enable_examview</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>show_examview_html</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>show_examview_pdf</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>enable_archiving</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>sign_submission</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>char_selector_availability</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>char_selector_definition</fieldlabel> <fieldentry/> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>skill_service</fieldlabel> <fieldentry>0</fieldentry> </qtimetadadatafield> </pre>	<p>fieldlabel -> customstyle Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> shuffle_questions Fragen mischen: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> processing_time Bearbeitungsdauer begrenzen: Bearbeitungsdauer im Format „hh:mm:ss“ hh = Stunden, mm = Minute, ss = Sekunden Die Anzeige im ILIAS System ist in Minuten angegeben und wird beim Export auf dieses Format konvertiert.</p> <p>fieldlabel -> enable_examview Übersicht gegebener Antworten: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> show_examview_html Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> show_examview_pdf Übersicht gegebener Antworten: PDF-Download: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> enable_archiving Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> sign_submission Digitale Signatur: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> char_selector_availability Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> char_selector_definition Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> skill_service Eintrag konnte im ILIAS nicht zugeordnet werden</p>
--	---

Tabelle 14: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 7/10

<pre> <qtimetadefinition> <fieldlabel>result_tax_filters</fieldlabel> <fieldentry>a:0:{}</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>show_grading_status</fieldlabel> <fieldentry>1</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>show_grading_mark</fieldlabel> <fieldentry>1</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>starting_time</fieldlabel> <fieldentry>P2021Y6M1DT15H50M0S</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>ending_time</fieldlabel> <fieldentry>P2021Y6M9DT17H0M0S</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>activation_limited</fieldlabel> <fieldentry>0</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>activation_start_time</fieldlabel> <fieldentry>0</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>activation_end_time</fieldlabel> <fieldentry>0</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>activation_visibility</fieldlabel> <fieldentry>0</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>autosave</fieldlabel> <fieldentry>1</fieldentry> </qtimetadefinition> <qtimetadefinition> <fieldlabel>autosave_ival</fieldlabel> <fieldentry>30000</fieldentry> </qtimetadefinition> </pre>	<p>fieldlabel -> result_tax_filters Eintrag konnte im ILIAS nicht gefunden werden</p> <p>fieldlabel -> show_grading_status Auswertung -> Teilnehmer sehen Testergebnisse: „Bestanden“ / „Nicht bestanden anzeigen“: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> show_grading_mark Auswertung -> Teilnehmer sehen Testergebnisse: Resultierende Note anzeigen: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> starting_time Startzeit des Tests. Hier: „01.06.21 – 15:50 Uhr“</p> <p>fieldlabel -> ending_time Ende des Tests. Hier: „09.06.21 – 17:00 Uhr“</p> <p>fieldlabel -> activation_limited Zeitlich begrenzte Verfügbarkeit: „Aktiviert“ – Dieser Wert kann nicht exportiert/importiert werden.</p> <p>fieldlabel -> activation_start_time Zeitlich begrenzte Verfügbarkeit: „Zeitraum Anfang“ – Dieser Wert kann nicht exportiert/importiert werden.</p> <p>fieldlabel -> activation_end_time Zeitlich begrenzte Verfügbarkeit: „Zeitraum Ende“ – Dieser Wert kann nicht exportiert/importiert werden.</p> <p>fieldlabel -> activation_visibility Zeitlich begrenzte Verfügbarkeit: „Immer sichtbar“ – Dieser Wert kann nicht exportiert/importiert werden.</p> <p>fieldlabel -> autosave Automatisches Speichern: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> autosave_ival Automatisches Speichern: Speicherintervall (in ms)</p>
--	--

Tabelle 15: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen –8/10

<pre> <qtimetadafield> <fieldlabel>offer_question_hints</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>instant_feedback_specific</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>instant_feedback_answer_fixation</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>obligations_enabled</fieldlabel> <fieldentry>0</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>enable_processing_time</fieldlabel> <fieldentry>1</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>mark_step_0</fieldlabel> <fieldentry><short>nicht bestanden</short><official>nicht bestanden</official> <percentage>0.00</percentage><passed>0</passed></fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>mark_step_1</fieldlabel> <fieldentry><short>bestanden</short><official>bestanden</official> <percentage>70.00</percentage><passed>1</passed></fieldentry> </qtimetadafield> </qtimetadafield> </pre>	<p>fieldlabel -> offer_question_hints Lösungshinweise: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> instant_feedback_specific Direkte Rückmeldung: Inhalte der Rückmeldung: „0“ – Deaktiviert „1“ – „Unterschiedliche Rückmeldungen pro gegebener Antwort“ aktiviert</p> <p>fieldlabel -> instant_feedback_answer_fixation Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>fieldlabel -> obligations_enabled Verpflichtende Fragen: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> enable_processing_time Bearbeitungsdauer begrenzen: „0“ – Deaktiviert „1“ – Aktiviert</p> <p>fieldlabel -> mark_step_0 Notenschema: Notenstufe 0 – „Kurzbezeichnung“, „Offizielle Bezeichnung“, Mindestprozentsatz“, „Bestanden“</p> <p>Es können verschiedene Notenstufen eingerichtet werden. Die Eingaben für „Kurzbezeichnung“, „Offizielle Bezeichnung“ und „Mindestprozentsatz“ sind variable Text Eingaben. Nicht bestanden: „Bestanden“ – „0“ Bestanden: „Bestanden“ – „1“</p> <p>fieldlabel -> mark_step_1 Notenschema: Notenstufe 1 – „Kurzbezeichnung“, „Offizielle Bezeichnung“, Mindestprozentsatz“, „Bestanden“</p>
---	---

Tabelle 16: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 9/10

<pre> <objectives> <material> <mattext texttype="text/xhtml"><p>Der Test basiert auf einer alten Klausuraufgabe. Sie können testen, ob Sie bereits gut mitkommen im Stoff oder nicht. </p> <p>Legen Sie sich zur Bearbeitung Ihren Taschenrechner, Ihre handgeschriebene Formelsammlung, Papier und Stift bereit. Wenn Sie eine Klausur-ähnliche Situation erzeugen wollen, nutzen Sie bitte keine Bücher und auch keine Internetseiten oder Mitstudierende. Nur dann bekommen Sie auch das Feedback, darüber was Sie schon können. </p> <p>Klicken Sie auf Start, sobald Sie den Test beginnen. Sie können den Test 5 Male durchführen. Die erfolgreiche Teilnahme zählt für Bonuspunkte wie im Skript beschrieben. Sobald die Frist für die Durchführung des Tests abgelaufen ist, wird die Original-Klausur-Aufgabe mitsamt Lösungen in den Tutorien vorgestellt und Sie bekommen die Gelegenheit, Fragen zu stellen. </p> </mattext> </material> </objectives> <assessmentcontrol solutionswitch="Yes"/> <presentation_material> <flow_mat> <material> <mattext texttype="text/xhtml"><p>Sie haben den Test absolviert. Wenn Sie mindestens 70% der Punkte erreicht haben, dann wird dieser Test bei den Zwischentest-Bonuspunkten berücksichtigt. </p> <p>Notieren Sie sich bitte, welche offenen Fragen Sie haben. Suchen Sie in den Vorlesungsunterlagen und in den Lehrbüchern nach Erklärungen oder bitten Sie Ihre Kollegen und die Tutoren darum. </p> <p>Sie haben insgesamt 5 Versuche für den Test. Zur Übung können Sie den Test also bis zu fünf Male wiederholen. </p> <p>In den Tutorien haben Sie die Möglichkeiten nach Lösungen für die Aufgaben zu fragen. </p> </mattext> </material> </flow_mat> </presentation_material> </pre>	<p>objectives -> ... -> mattext Allgemeine Einstellungen -> Einleitung</p> <p>Dieser Text wird als Einleitung für einen ILIAS-Test dargestellt.</p> <p>assessmentcontrol -> solutionswitch Eintrag konnte nicht in den ILIAS Einstellungen gefunden werden.</p> <p>presentation_material -> ... -> mattext Allgemeine Einstellungen -> Abschließende Bemerkung</p> <p>Dieser Text wird als abschließende Bemerkung für einen ILIAS-Test dargestellt.</p>
--	--

Tabelle 17: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 10/10

4.3.3 Aufbau der qpl.xml Dateien

<pre> <!DOCTYPE Test SYSTEM "http://www.ilias.uni-koeln.de/download/dtd/ilias_co.dtd"> <!--Export of ILIAS Test Questionpool 2423470 of installation .--> <ContentObject Type="Questionpool_Test"> <MetaData> <General Structure="Hierarchical"> <Identifier Catalog="ILIAS" Entry="il_0_qpl_2423470"/> <Title Language="de">Aufgabensammlung</Title> <Language Language="de"/> <Description Language="de"/> <Keyword Language="de"/> </General> </MetaData> <Settings> <ShowTaxonomies>0</ShowTaxonomies> <NavTaxonomy>0</NavTaxonomy> <SkillService>0</SkillService> </Settings> <PageObject> <PageContent> <Question QRef="il_0_qst_677964"/> </PageContent> </PageObject> <PageObject> <PageContent> <Question QRef="il_0_qst_677965"/> </PageContent> </PageObject> <QuestionSkillAssignments> <TriggerQuestion Id="677964"/> <TriggerQuestion Id="677965"/> </QuestionSkillAssignments> </ContentObject> </pre>	<p>ContentObject Type „Questionpool_Test“: Ist ein fixer Eintrag der für jeden Fragenpool oder –test identisch ist</p> <p>Identifier Catalog „ILIAS“: Fixer Eintrag</p> <p>Identifier Entry „il_0_qpl_2423470“: Spiegelt den Namen des Fragenpools wider. Muss identisch mit dem Dateinamen sein!</p> <ul style="list-style-type: none"> Z.B.: Dateiname des Fragenpools: „1624027320__0__qpl_2423470“ <p>Title: Gibt den Namen des Fragenpools an</p> <p>Settings: ShowTaxonomies, NavTaxonomy, SkillService: Haben als Eintrag standardmäßig „0“. Hat keine sichtbaren Auswirkungen auf das Verhalten des Fragenpools.</p> <p>Question QRef = „il_0_qst_677964“: Gibt die ID der jeweiligen Frage an. Alle Fragen die im Fragentest oder –pool enthalten sind müssen hier als ID auftauchen. Die eigentliche Frage (mit der ID) ist in der QTI.xml enthalten</p> <p>TriggerQuestion ID = „677964“: Hier wird nochmals die ID abgelegt. Werden die Fragen ID’s nicht komplett abgebildet, kann es zu Fehlern beim ILIAS Import führen</p>
---	--

Tabelle 18: Darstellung von XML-Datei: qpl.xml (für einen Fragenpool)

4.3.4 Aufbau der qti.xml Dateien

4.3.4.1 Darstellung der XML Struktur für den Fragentyp „Formelfrage“

Formelfrage spezifischer Header

<p>XML-Struktur für Fragentyp „Formelfrage“ ohne Einsatz von „Einheiten“ Zur besseren Lesbarkeit wurden die Einträge für die Variable (\$v1), für das Ergebnis (\$r1) und der Fragentext (material-> mattext) mit Leerzeichen versehen</p> <pre> <questestinterop> <item ident="il_0_qst_400000" title="01.2.01 Elektrischer Strom"> <qticomment>Elektrischen Strom aus Ladung pro Zeit berechnen. </qticomment> <duration>POYOMODT1H30M0S</duration> <itemmetadata> <qtimetadadatafield> <fieldlabel>ILIAS_VERSION</fieldlabel> <fieldentry>5.4.10 2020-03-04</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>QUESTIONTYPE</fieldlabel> <fieldentry>assFormulaQuestion</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>AUTHOR</fieldlabel> <fieldentry /> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>points</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> </itemmetadata> </item> </questestinterop> </pre>	<p>questestinterop: Jede XML –Baumstruktur (egal ob Fragen-Test oder –Pool) beginnt mit dem XML-Eintrag „questestinterop“</p> <p>item: ident: Gibt die ID der Frage an. Hier kann eine beliebige 6-stellige Nummer eingetragen werden.</p> <p>item: title: Gibt den Fragentitel an. Der Fragentitel darf in einem Aufgabenpool nur 1x vorkommen.</p> <p>qticomment: Gibt die Fragenbeschreibung an.</p> <p>duration: Gibt die Zeit zur Bearbeitung für die Frage an</p> <p>ILIAS_VERSION: 5.4.10 2020-03-04: Wird eine XML von ILIAS exportiert, wird immer die verwendete ILIAS-Version mit angegeben. Für den Import in ILIAS wurde in die XML eine „feste“ ILIAS-Version im Programmcode hinterlegt. Unter Umständen muss diese Nummer in Zukunft angepasst werden.</p> <p>QUESTIONTYPE: Gibt den Fragentyp an. „assFormulaQuestion“ steht im ILIAS für eine „Formelfrage“</p> <p>AUTHOR: Gibt den Fragen-Autor an.</p> <p>points: Gibt die Punktzahl für eine richtige Beantwortung der Frage an</p>
---	---

Tabelle 19: Darstellung der XML-Fragenstruktur für eine Formelfrage – Spezifischer Header – 1/4

Formelfrage: Definition für Variable

<pre> <qtimetadadatafield> <fieldlabel>\$v1</fieldlabel> <fieldentry>a:6:{ s:9:"precision";i:0; s:12:"intprecision";s:1:"1"; s:8:"rangemin";d:10; s:8:"rangemax";d:60; s:4:"unit";s:0:""; s:9:"unitvalue";s:0:""; } </fieldentry> <!-- Beispiel für einen Eintrag mit Einheiten: --> <!-- <fieldentry>a:6{ --> <!-- s:9:"precision";d:0; s:12:"intprecision";s:1:"1"; --> <!-- s:8:"rangemin";d:10; s:8:"rangemax";d:60; --> <!-- s:4:"unit";s:2:"mV"; s:9:"unitvalue";s:3:"141"; --> <!-- } --> <!-- </fieldentry> --> </qtimetadadatafield> </pre>	<p>\$v1 – Definiert die in der Frage verwendete Variable. Aufgeschlüsselt stehen die Begriffe für folgendes:</p> <p>a:6: -- In der nachfolgenden Klammer werden 6 Einträge behandelt.</p> <ul style="list-style-type: none"> „precision“ -- Präzision (Anzahl der Nachkommastellen) „interprecision“ -- Teilbar durch (Bei einer Präzision von 0, legt dieser Wert fest durch welche ganze Zahl die erzeugte Variable teilbar sein muss) „rangemin“ -- Bereich Minimum (Legt den kleinstmöglichen Wert für die Variable fest) „rangemax“ -- Bereich Maximum (Legt den höchstmöglichen Wert für die Variable fest) „unit“ -- Einheit (Legt die Einheit der Variable fest) „unitvalue“ -- Einheit-Wert <p>Aufschlüsselung der Einträge für Variablen</p> <p>z.B. s:8:"rangemin";d:10;</p> <ul style="list-style-type: none"> s: – Nachfolgender Eintrag (hier „rangemin“) ist ein String/Text 8: Nachfolgender Eintrag (hier „rangemin“) ist 8 Zeichen lang d: Nachfolgender Eintrag ist eine ganze Zahl <p>Wichtig!</p> <p>Unter z.B. „rangemin“ könnte statt „d“ (für eine feste Zahl) auch ein „s“ für einen Text eingegeben werden. Dadurch kann als rangemin auch z.B. „10*10⁻⁶“ eingetragen werden. Das Ergebnis wird dann von ILIAS berechnet. Der Eintrag muss dann folgender Maßen abgeändert werden:</p> <p>s:8:"rangemin";s:6:"10*\$v2";</p> <p>Unit und Unitvalue für Variablen</p> <p>In der derzeitigen ILIAS-Version werden keine globalen Einheiten unterstützt. Es ist zwar möglich lokale Einheiten zu definieren:</p> <ul style="list-style-type: none"> Definition Basis = 1, „m“ Definition km = 1000* Basis, „km“ Definition mm = (1/1000)*Basis, „mm“ <p>Jedoch muss im ILIAS für das Ergebnis die zu erwartende Einheit auch angegeben werden d.h. das Ergebnis soll zwingend als z.B. „m“ angegeben werden. Dadurch ist es nicht möglich, dass das Ergebnis von Student A (0,1 km) das gleiche Ergebnis erzielt, wie die Antwort von Student B (100m). Durch die vorherige Festlegung des Ergebnisses auf „m“ wird 0,1 km im ILIAS als falsch betrachtet!</p> <p>Die Definition von Einheiten über die XML-Datei läuft wie folgt ab:</p> <p>Für die Variable wird unter dem Punkt „Unit“ eine eigene Einheit eingetragen (hier ist keine Beschränkung gegeben). Anschließend muss für diese neue Einheit eine ID zugeordnet werden (eine 3-stellige Zahl nach Belieben). Der Name der Einheit und die ID müssen unter dem „Ergebnis“ (\$r1) wieder auftauchen.</p>
---	---

Tabelle 20: Darstellung der XML-Fragenstruktur für eine Formelfrage – Definition für Variablen – 2/4.

Formelfrage: Definition für Ergebnis

<pre> <qtimetadatatfield> <fieldlabel>\$r1</fieldlabel> <fieldentry>a:10:{ s:9:"precision";i:2; s:9:"tolerance";s:1:"5"; s:8:"rangemin";s:1:"0"; s:8:"rangemax";s:6:"100000"; s:6:"points";s:1:"1"; s:7:"formula";s:26:"((\$v2*0.001)/\$v1)/0.000001"; s:6:"rating";s:0:""; s:4:"unit";s:0:""; s:9:"unitvalue";s:0:""; s:11:"resultunits";a:0:{}} </fieldentry> <!-- Beispiel für einen Eintrag mit Einheiten: --> <!-- <fieldentry>a:10:{ --> <!-- s:9:"precision";i:2; s:9:"tolerance";s:1:"5"; --> <!-- s:8:"rangemin";s:1:"0"; s:8:"rangemax";s:6:"100000"; --> <!-- s:6:"points";s:1:"1"; s:7:"formula";s:26:"((\$v2*0.001)/\$v1)/0.000001"; --> <!-- s:6:"rating";s:0:""; s:4:"unit";s:2:"mV"; --> <!-- s:9:"unitvalue";s:3:"141"; s:11:"resultunits"; a:7:{ --> <!-- i:0;a:2:{s:4:"unit";s:1:"H";s:9:"unitvalue";s:3:"125";} --> <!-- i:1;a:2:{s:4:"unit";s:2:"mH";s:9:"unitvalue";s:3:"126";} --> <!-- i:2;a:2:{s:4:"unit";s:3:"μH";s:9:"unitvalue";s:3:"127";} --> <!-- ... --> <!-- i:6;a:2:{s:4:"unit";s:1:"F";s:9:"unitvalue";s:3:"131";} --> <!-- } --> <!-- </fieldentry> --> </qtimetadatatfield> </pre>	<p>\$r1 – Definiert das in der Frage verwendete Ergebnis. Aufgeschlüsselt stehen die Begriffe für folgendes:</p> <p>a:10: -- In der nachfolgenden Klammer werden 10 Einträge behandelt.</p> <ul style="list-style-type: none"> „precision“ -- Präzision (Anzahl der Nachkommastellen) „tolerance“ -- Toleranz in % (Legt die erlaubte Abweichung vom Ergebniswert fest.) „rangemin“ -- Bereich Minimum (Legt den kleinstmöglichen Wert für das Ergebnis fest) „rangemax“ -- Bereich Maximum (Legt den höchstmöglichen Wert für das Ergebnis fest) „points“ -- Punkte (Legt die erreichbare Punktzahl für die Frage fest) „formula“ -- Formel (Legt die Formel fest, mit der das Ergebnis von ILIAS berechnet werden kann) „rating“ -- Eintrag konnte im ILIAS nicht zugeordnet werden „unit“ -- Einheit (Legt die Einheit der Variable fest) „unitvalue“ -- Einheit-Wert „resultunits“ -- Auswählbare Einheiten <p>Beschreibung der einzelnen Zeichen, wie unter \$v1. Rangemin und rangemax sind, im Gegensatz zu der Definition bei den Variablen, bei den Ergebnissen standardmäßig mit einem „s“ (Text) definiert:</p> <ul style="list-style-type: none"> s:8:"rangemin";s:1:"0"; <p>Unit, Unitvalue und Resultunits für Ergebnisse</p> <p>Die Definition von Einheiten über die XML-Datei läuft wie folgt ab:</p> <p>Für das Ergebnis wird unter dem Punkt „Unit“ eine eigene Einheit eingetragen (hier ist keine Beschränkung gegeben). Anschließend muss für diese neue Einheit eine ID zugeordnet werden (eine 3-stellige Zahl nach Belieben) Dazu wird in der Aufgabe automatisch eine Auswahlbox erzeugt, in welcher die richtige Einheit ausgewählt werden kann. Das Ergebnis muss später in der Aufgabe dann in dieser definierten Einheit angegeben werden.</p> <p>Welche Einträge in dieser Auswahlbox zur Verfügung stehen, wird unter „resultunits“ definiert. Alle Einträge die hier definiert werden, können später bei der Aufgabe aus der Auswahlbox ausgewählt werden.</p> <p>Wichtig!</p> <p>Eine automatische Umrechnung seitens ILIAS ist, wie zuvor erwähnt, nicht möglich.</p> <p>„100m“ sind nicht gleich „0,1km“!</p>
--	--

Tabelle 21: Darstellung der XML-Fragenstruktur für eine Formelfrage – Definition für Ergebnisse – 3/4

Formelfrage: Beschreibung Fragentext

<pre> <fieldlabel>additional_cont_edit_mode</fieldlabel> <fieldentry>default</fieldentry> </qtimetadafield> <qtimetadafield> <fieldlabel>externalId</fieldlabel> <fieldentry>5ea15be69c1e96.43933468</fieldentry> </qtimetadafield> </qtimetadadata> </itemmetadata> <presentation label="01.2.01 Elektrischer Strom"> <flow> <material> <mattext texttype="text/html"><p></p> <p> Durch einen elektrischen Leiter bewegen sich in einer Zeit <i>t</i> = \$v1 s Ladungsträger mit eine Ladung <i>q</i>= \$v2 mC. Wie groß ist der elektrische Strom <i>I</i> im Leiter? </p> <p> <i>I</i> = \$r1 &mu;A</p></mattext> <matimage label="il_0_mob_30599" uri="objects/il_0_mob_30599/figurehNbH3w.png"> </material> </flow> </presentation> </item> </questestinterop> </pre>	<p>Additional_cont_edit_mode: Eintrag konnte im ILIAS nicht zugeordnet werden</p> <p>External_ID: Ist für die Erstellung von Fragen nicht von Bedeutung. Kann in jeder Frage gleich sein.</p> <p>Presentation: label Gibt den Namen der Frage an. Dieser Name muss der Gleiche sein, wie unter item → title</p> <p>material: mattext: texttype: Hier wird der Text-Typ festgelegt. Muss nicht angepasst werden.</p> <p>material: mattext: Hier steht der Fragen-Text. Wird ein Bild im Fragen-Text verwendet, steht in dem Fragen-Text zusätzlich das Bild. Die Position dieses Eintrages, bestimmt die Position im Fragentext. Steht der Eintrag gleich zu Beginn, wird das Bild an oberster Stelle stehen. Bild-Eintrag (Bildhöhe in Pixeln, Bildquelle, Bildbreite in Pixeln): <p></p></p> <p>Einträge werden immer in <p></p> eingebettet. Typische HTML Befehle werden beim Import in ILIAS entsprechend umgesetzt z.B.</p> <ul style="list-style-type: none"> • &mu;A → μA • <i>Test</i> → Test <p>material: matimage: label: Hier steht der Ordnername des Bildes. Soll das Bild verwendet werden, muss der Ordnername im Fragen-Text unter „src“ der Gleiche sein.</p> <p>material: matimage: uri: Gibt den Pfad zum Ordnernamen des verwendeten Bildes an. Das Bild befindet sich immer im Ordner ../objects/il_mob_xxxxx</p>
---	--

Tabelle 22: Darstellung der XML-Fragenstruktur für eine Formelfrage – Beschreibung Fragen-Text – 4/4

4.3.4.2 Darstellung der XML Struktur für den Fragentyp „Singlechoice“

Singlechoice spezifischer Header

<pre> <questestinterop> <item ident="il_0_qst_400000" title="Test_Aufgabe"> <qtcomment /> <duration>POYOM0DT23H0M0S</duration> <itemmetadata> <qtimetadadata> <qtimetadadatafield /> <qtimetadadatafield> <fieldlabel>ILIAS_VERSION</fieldlabel> <fieldentry>5.4.14 2020-07-31</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>QUESTIONTYPE</fieldlabel> <fieldentry>SINGLE CHOICE QUESTION</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>AUTHOR</fieldlabel> <fieldentry>Panteleit</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>additional_cont_edit_mode</fieldlabel> <fieldentry>default</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>externalId</fieldlabel> <fieldentry>5f11d3ed9af3e5.53678796</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>thumb_size</fieldlabel> <fieldentry>300</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>feedback_setting</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>singleline</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> </qtimetadadata> </itemmetadata> </item> </questestinterop> </pre>	<p>Item -> ident: Gibt die ID der Frage an. Hier kann eine beliebige 6-stellige Nummer eingetragen werden. „il_0_qst_“ darf nicht verändert werden! Eine Änderung der ID muss auch in der qpl.xml übertragen werden.</p> <p>Item -> title: Gibt den Fragentitel an. Der Fragentitel darf in einem Aufgabenpool nur 1x vorkommen.</p> <p>qtcomment: Gibt die Fragenbeschreibung an.</p> <p>duration: Gibt die Zeit zur Bearbeitung für die Frage an</p> <p>ILIAS_VERSION: Wird eine XML von ILIAS exportiert, wird immer die verwendete ILIAS-Version mit angegeben. Für den Import in ILIAS wurde in die XML eine „feste“ ILIAS-Version im Programmcode hinterlegt. Unter Umständen muss diese Nummer in Zukunft angepasst werden.</p> <p>QUESTIONTYPE: Gibt den Fragentyp an. „SINGLE CHOICE QUESTION“ steht im ILIAS für eine „Singlechoice“-Frage</p> <p>AUTHOR: Gibt den Fragen-Autor an.</p> <p>additional_cont_edit_mode: <i>Eintrag konnte im ILIAS nicht zugeordnet werden</i></p> <p>externalId: Ist für die Erstellung von Fragen nicht von Bedeutung. Kann in jeder Frage gleich sein.</p> <p>thumb_size: Jeder Antwort kann ein Bild angefügt werden. „Thumb_size“ bestimmt die Größe des Bildes in den Antworten (in Pixeln).</p> <p>feedback_setting: <i>Eintrag konnte im ILIAS nicht zugeordnet werden</i></p> <p>singleline: Gibt an ob 1- oder 2-zeilige Antworten zugelassen sind.</p>
--	---

Tabelle 23: Darstellung der XML-Fragenstruktur für eine Singlechoice Frage – Spezifischer Header – 1/3

Singlechoice: Definition - Antworten

<pre> <presentation label="Test_Aufgabe"> <flow> <material> <mattext texttype="text/html"><p>Das ist ein Beispiel-Text</p></mattext> </material> <response_lid ident="MCSR" rcardinality="Single"> <render_choice shuffle="Yes"> <!-- ----- Singlechoice Antwort 1 ----- --> <response_label ident="0"> <material> <mattext texttype="text/plain">Bild1</mattext> <matimage imagtype="image/png" label="il_0_mob_30599.png" embedded="base64"> iVBORw0KGgoAAAANSUUEUgAAAYAAAAEs=</matimage> </material> </response_label> <!-- ----- Singlechoice Antwort 2 ----- --> <response_label ident="1"> <material> <mattext texttype="text/plain">Bild2</mattext> <matimage imagtype="image/png" label="il_0_mob_30600.png" embedded="base64"> iVBORw0KGgoAAAANSUUEUgAAAYAAAAEs=</matimage> </material> </response_label> </render_choice> </response_lid> </flow> </presentation> </pre>	<p>presentation -> label: Hier steht der Fragentitel. Es muss der gleiche Eintrag wie unter „item ident“ sein</p> <p>presentation -> flow -> material -> mattext -> texttype: Gibt die Formatierung des nachfolgenden Textes an. Kann als „fix“ angesehen werden.</p> <p>presentation -> flow -> material -> mattext -> Text: Beschreibt den Fragen-Text (Haupt-Text). Muss zwingend in „<p> </p>“ eingeklammert sein!</p> <p>response_lid -> Ident & rcardinality: „MCSR“ und „Single“ stehen für eine „Singlechoice“ Antwort.</p> <p>response_lid -> render_choice -> shuffle: Gibt an ob die Reihenfolge der Antworten gemischt werden sollen. Bei der Beantwortung der Frage im Test, kann die definierte „Antwort 1“ auch an einer anderen Stelle stehen</p> <p>response_lid -> ... -> response_label -> ident: Gibt die Nummer der Antwort. Wird für jede Antwort inkrementiert (beginnend bei „0“)</p> <p>response_lid -> ... -> mattext -> texttype: Gibt die Formatierung des nachfolgenden Textes an. Kann als „fix“ angesehen werden.</p> <p>response_lid -> ... -> mattext -> Text: Spiegelt den Antwort-Text wider.</p> <p>response_lid -> ... -> matimage -> imagtype: Gibt das Format des Bildes an. Es muss zwingend das richtige Format angegeben werden!</p> <p>response_lid -> ... -> matimage -> label: Gibt den gesamten Namen des Bildes an (mit Format)</p> <p>response_lid -> ... -> matimage -> embedded: Die verwendeten Bilder, innerhalb einer Singlechoice Antwort, sind „Base64“ codiert. Das codierte Bild wird als „String“ dargestellt</p>
---	--

Tabelle 24: Darstellung der XML-Fragenstruktur für eine Singlechoice Frage – Definition für Antworten – 2/3

Singlechoice: Definition – Punktevergabe und Rückmeldung für Antworten

<pre> <resprocessing> <!-- Punkte für Singlechoice Antwort 1 ----- --> <respcondition continue="Yes"> <conditionvar> <varequal respident="MCSR">0</varequal> </conditionvar> <setvar action="Add">1</setvar> <displayfeedback feedbacktype="Response" linkrefid="response_0" /> </respcondition> <!-- Punkte für Singlechoice Antwort 2 ----- --> <respcondition continue="Yes"> <conditionvar> <varequal respident="MCSR">1</varequal> </conditionvar> <setvar action="Add">1</setvar> <displayfeedback feedbacktype="Response" linkrefid="response_1" /> </respcondition> </resprocessing> <!-- Rückmeldung für Singlechoice Antwort 1 ----- --> <itemfeedback ident="response_0" view="All"> <flow_mat> <material> <mattext texttype="text/plain" /> </material> </flow_mat> </itemfeedback> <!-- Rückmeldung für Singlechoice Antwort 2 ----- --> <itemfeedback ident="response_1" view="All"> <flow_mat> <material> <mattext texttype="text/plain" /> </material> </flow_mat> </itemfeedback> </item> </questestinterop> </pre>	<p>respcondition -> continue: Jede Singlechoice Antwort beginnt mit einem „continue=“Yes“ Eintrag. Gibt an ob eine Antwort zur Verfügung steht.</p> <p>varequal -> respident: „MCSR“ steht für eine Singlechoice Frage</p> <p>varequal -> Text: Hier steht die „Id“ der Antwort. Wird für jede Antwort inkrementiert (beginnend bei „0“)</p> <p>setvar -> action: „Add“ addiert die nachfolgenden Punkte bei einer Auswahl dieser Antwort.</p> <p>setvar -> Text: Gibt die Anzahl der Punkte für die Antwort an.</p> <p>itemfeedback -> ident: Gibt die „ID“ der Antwort an. Wird für jede inkrementiert (beginnend bei „response_0“)</p> <p>itemfeedback -> view: Standardeintrag „All“. Verändert nichts im ILIAS und muss nicht berücksichtigt werden.</p> <p>itemfeedback -> ... -> mattext -> texttype: Format der Antwort</p> <p>itemfeedback -> ... -> mattext -> text: Gibt den Text für eine Rückmeldung einer ausgewählten Antwort an. Wird hier ein Text eingetragen muss dieser zwingend mit „<p> </p>“ eingeklammert werden!</p>
---	---

Tabelle 25: Darstellung der XML-Fragenstruktur für eine Singlechoice Frage – Definition Punktevergabe und Rückmeldung für Antworten – 3/3

4.3.4.3 Darstellung der XML Struktur für den Fragentyp „Multiplechoice“

Multiplechoice spezifischer Header

<pre> <questestinterop> <item ident="il_0_qst_400000" title="Fragen Test in MC"> <qtcomment /> <duration>POYOM0DT23H0M0S</duration> <itemmetadata> <qtimetadadata> <qtimetadadatafield /> <qtimetadadatafield> <fieldlabel>ILIAS_VERSION</fieldlabel> <fieldentry>5.4.14 2020-07-31</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>QUESTIONTYPE</fieldlabel> <fieldentry>MULTIPLE CHOICE QUESTION</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>AUTHOR</fieldlabel> <fieldentry>Panteleit</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>additional_cont_edit_mode</fieldlabel> <fieldentry>default</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>externalId</fieldlabel> <fieldentry>5f11d3ed9af3e5.53678796</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>thumb_size</fieldlabel> <fieldentry>300</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>feedback_setting</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>singleline</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> </qtimetadadata> </itemmetadata> </item> </questestinterop> </pre>	<p>Item -> ident: Gibt die ID der Frage an. Hier kann eine beliebige 6-stellige Nummer eingetragen werden. „il_0_qst_“ darf nicht verändert werden! Eine Änderung der ID muss auch in der qpl.xml übertragen werden.</p> <p>Item -> title: Gibt den Fragentitel an. Der Fragentitel darf in einem Aufgabenpool nur 1x vorkommen.</p> <p>qtcomment: Gibt die Fragenbeschreibung an.</p> <p>duration: Gibt die Zeit zur Bearbeitung für die Frage an</p> <p>ILIAS_VERSION: Wird eine XML von ILIAS exportiert, wird immer die verwendete ILIAS-Version mit angegeben. Für den Import in ILIAS wurde in die XML eine „feste“ ILIAS-Version im Programmcode hinterlegt. Unter Umständen muss diese Nummer in Zukunft angepasst werden.</p> <p>QUESTIONTYPE: Gibt den Fragentyp an. „MULTIPLE CHOICE QUESTION“ steht im ILIAS für eine „Multiplechoice“-Frage</p> <p>AUTHOR: Gibt den Fragen-Autor an.</p> <p>additional_cont_edit_mode: <i>Eintrag konnte nicht im ILIAS gefunden werden</i></p> <p>externalId: Ist für die Erstellung von Fragen nicht von Bedeutung. Kann in jeder Frage gleich sein.</p> <p>thumb_size: Jeder Antwort kann ein Bild angefügt werden. „Thumb_size“ bestimmt die Größe des Bildes in den Antworten (in Pixeln).</p> <p>feedback_setting: <i>Eintrag konnte nicht im ILIAS gefunden werden</i></p> <p>singleline: Gibt an ob 1- oder 2-zeilige Antworten zugelassen sind.</p>
---	---

Tabelle 26: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Spezifischer Header – 1/4

Multiplechoice: Definition - Antworten

<pre> <presentation label="Test_Aufgabe"> <flow> <material> <mattext texttype="text/html"><p>Das ist ein Beispiel-Text</p></mattext> </material> <response_lid ident="MCMR" rcardinality="Multiple"> <render_choice shuffle="Yes"> <!-- ----- Multiplechoice Antwort 1 ----- --> <response_label ident="0"> <material> <mattext texttype="text/plain">Bild1</mattext> <matimage imagtype="image/png" label="il_0_mob_30599.png" embedded="base64"> iVBORw0KGgoAAAANSUgAAUgAAAEs=</matimage> </material> </response_label> <!-- ----- Multiplechoice Antwort 2 ----- --> <response_label ident="1"> <material> <mattext texttype="text/plain">Bild2</mattext> <matimage imagtype="image/png" label="il_0_mob_30600.png" embedded="base64"> iVBORw0KGgoAAAANSUgAAUgAAAEs=</matimage> </material> </response_label> </render_choice> </response_lid> </flow> </presentation> </pre>	<p>presentation -> label: Hier steht der Fragentitel. Es muss der gleiche Eintrag wie unter „item ident“ sein</p> <p>presentation -> flow -> material -> mattext -> texttype: Gibt die Formatierung des nachfolgenden Textes an. Kann als „fix“ angesehen werden.</p> <p>presentation -> flow -> material -> mattext -> Text: Beschreibt den Fragen-Text (Haupt-Text). Muss zwingend in „<p> </p>“ eingeklammert sein!</p> <p>response_lid -> Ident & rcardinality: „MCMR“ und „Multiple“ stehen für eine „Multiplechoice“ Antwort.</p> <p>response_lid -> render_choice -> shuffle: Gibt an ob die Reihenfolge der Antworten gemischt werden sollen. Bei der Beantwortung der Frage im Test, kann die definierte „Antwort 1“ auch an einer anderen Stelle stehen</p> <p>response_lid -> ... -> response_label -> ident: Gibt die Nummer der Antwort. Wird für jede Antwort inkrementiert (beginnend bei „0“)</p> <p>response_lid -> ... -> mattext -> texttype: Gibt die Formatierung des nachfolgenden Textes an. Kann als „fix“ angesehen werden.</p> <p>response_lid -> ... -> mattext -> Text: Spiegelt den Antwort-Text wider.</p> <p>response_lid -> ... -> matimage -> imagtype: Gibt das Format des Bildes an. Es muss zwingend das richtige Format angegeben werden!</p> <p>response_lid -> ... -> matimage -> label: Gibt den gesamten Namen des Bildes an (mit Format)</p> <p>response_lid -> ... -> matimage -> embedded: Die verwendeten Bilder, innerhalb einer Multiplechoice Antwort, sind „Base64“ codiert. Das codierte Bild wird als „String“ dargestellt</p>
---	--

Tabelle 27: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Definition von Antworten – 2/4

Multiplechoice: Definition - Punktevergabe für Antworten

<pre> <resprocessing> <!-- ----- Punkte für Multiplechoice Antwort 1 ----- --> <respcondition continue="Yes"> <conditionvar> <varequal respident="MCMR">0</varequal> </conditionvar> <setvar action="Add">1</setvar> <displayfeedback feedbacktype="Response" linkrefid="response_0" /> </respcondition> <respcondition continue="Yes"> <conditionvar> <not> <varequal respident="MCMR">0</varequal> </not> </conditionvar> <setvar action="Add">0</setvar> </respcondition> <!-- ----- Punkte für Multiplechoice Antwort 2 ----- --> <respcondition continue="Yes"> <conditionvar> <varequal respident="MCMR">1</varequal> </conditionvar> <setvar action="Add">1</setvar> <displayfeedback feedbacktype="Response" linkrefid="response_1" /> </respcondition> <respcondition continue="Yes"> <conditionvar> <not> <varequal respident="MCMR">1</varequal> </not> </conditionvar> <setvar action="Add">0</setvar> </respcondition> </resprocessing> </pre>	<p>respcondition -> continue: Jede Multiplechoice Antwort beginnt mit einem „continue=“Yes“ Eintrag. Gibt an ob eine Antwort zur Verfügung steht.</p> <p>varequal -> respident: „MCMR“ steht für eine Multiplechoice Frage</p> <p>varequal -> Text: Hier steht die „Id“ der Antwort. Wird für jede Antwort inkrementiert (beginnend bei „0“)</p> <p>setvar -> action: „Add“ addiert die nachfolgenden Punkte, wenn diese Antwort <u>ausgewählt</u> wurde</p> <p>setvar -> Text: Gibt die Anzahl der Punkte für die Antwort an, wenn <u>ausgewählt</u> wurde</p> <p>not -> varequal -> respident: „MCMR“ steht für eine Multiplechoice Frage. „Not“ gibt den Eintrag für „Antwort <u>nicht</u> ausgewählt“</p> <p>not -> varequal -> Text: Jede Antwort einer Multiplechoice Frage beinhaltet den „Not“ Teil. Es handelt sich um die Punkte für eine Antwort, wenn diese <u>nicht</u> ausgewählt wurde. Eine Antwort ID kommt demnach immer zweimal vor.</p> <p>setvar -> action: „Add“ addiert die nachfolgenden Punkte, wenn diese Antwort <u>nicht ausgewählt</u> wurde</p> <p>setvar -> Text: Gibt die Anzahl der Punkte für die Antwort an, wenn <u>nicht ausgewählt</u> wurde.</p>
--	---

Tabelle 28: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Definition Punktevergabe für Antworten – 3/4

Multiplechoice: Definition –Rückmeldung für Antworten

<pre> <!-- ----- Rückmeldung für Multiplechoice Antwort 1 ----- --> <itemfeedback ident="response_0" view="All"> <flow_mat> <material> <mattext texttype="text/plain" /> </material> </flow_mat> </itemfeedback> <!-- ----- Rückmeldung für Multiplechoice Antwort 2 ----- --> <itemfeedback ident="response_1" view="All"> <flow_mat> <material> <mattext texttype="text/plain" /> </material> </flow_mat> </itemfeedback> </item> </questestinterop> </pre>	<p>itemfeedback -> ident: Gibt die „ID“ der Antwort an. Wird für jede inkrementiert (beginnend bei „response_0“)</p> <p>itemfeedback -> view: Standardeintrag „All“. Verändert nichts im ILIAS und muss nicht berücksichtigt werden.</p> <p>itemfeedback -> ... -> mattext -> texttype: Format der Antwort</p> <p>itemfeedback -> ... -> mattext -> text: Gibt den Text für eine Rückmeldung einer ausgewählten Antwort an. Wird hier ein Text eingetragen muss dieser zwingend mit „<p> </p>“ eingeklammert werden!</p>
---	--

Tabelle 29: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Definition Rückmeldung für Antworten – 4/4

4.3.4.4 Darstellung der XML Struktur für den Fragentyp „Zuordnungsfrage“

Zuordnungsfrage spezifischer Header

<pre> <questestinterop> <item id="il_0_qst_680436" title="MQ-Frage" maxattempts="0"> <qtcomment>Beschreibung für Zuordnungsfrage</qtcomment> <itemmetadata> <qtimetadadata> <qtimetadadatafield /> <qtimetadadatafield> <fieldlabel>ILIAS_VERSION</fieldlabel> <fieldentry>5.4.14 2020-07-31</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>QUESTIONTYPE</fieldlabel> <fieldentry>MATCHING QUESTION</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>AUTHOR</fieldlabel> <fieldentry>Panteleit</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>additional_cont_edit_mode</fieldlabel> <fieldentry>default</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>externalId</fieldlabel> <fieldentry>5f11d3ed9af3e5.53678796</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>shuffle</fieldlabel> <fieldentry>1</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>thumb_geometry</fieldlabel> <fieldentry>300</fieldentry> </qtimetadadatafield> <qtimetadadatafield> <fieldlabel>matching_mode</fieldlabel> <fieldentry>1:1</fieldentry> </qtimetadadatafield> </qtimetadadata> </itemmetadata> </item> </questestinterop> </pre>	<p>Item -> id: Gibt die ID der Frage an. Hier kann eine beliebige 6-stellige Nummer eingetragen werden. „il_0_qst_“ darf nicht verändert werden! Eine Änderung der ID muss auch in der qpl.xml übertragen werden.</p> <p>Item -> title: Gibt den Fragentitel an. Der Fragentitel darf in einem Aufgabenpool nur 1x vorkommen.</p> <p>qtcomment: Gibt die Fragenbeschreibung an.</p> <p>duration: Gibt die Zeit zur Bearbeitung für die Frage an.</p> <p>ILIAS_VERSION: Wird eine XML von ILIAS exportiert, wird immer die verwendete ILIAS-Version mit angegeben. Für den Import in ILIAS wurde in die XML eine „feste“ ILIAS-Version im Programmcode hinterlegt. Unter Umständen muss diese Nummer in Zukunft angepasst werden.</p> <p>QUESTIONTYPE: Gibt den Fragentyp an. „MATCHING QUESTION“ steht im ILIAS für eine „Zuordnungsfrage“-Frage.</p> <p>AUTHOR: Gibt den Fragen-Autor an.</p> <p>additional_cont_edit_mode: <i>Eintrag konnte nicht im ILIAS gefunden werden</i></p> <p>externalId: Ist für die Erstellung von Fragen nicht von Bedeutung. Kann in jeder Frage gleich sein.</p> <p>shuffle: Legt fest ob Definitionen/Terme gemischt werden sollen.</p> <ul style="list-style-type: none"> • „0“ – Nein • „1“ – Beides • „2“ – Nur Terme • „3“ – Nur Definitionen <p>thumb_geometry: Bestimmt die Größe der Bilder (Vorschaugröße) von Definitionen bzw. Termen (in Pixeln).</p> <p>matching_mode: Bestimmt den Zuordnungsmodus</p> <ul style="list-style-type: none"> • Ein Term kann einer Definition zugeordnet werden (1:1) • Ein oder mehrere Terme können einer oder mehreren Definitionen zugeordnet werden (n:n)
---	---

Tabelle 30: Darstellung der XML-Fragenstruktur für eine Zuordnungsfrage – Spezifischer Header – 1/3

Zuordnungsfrage: Definition – Definitionen und Terme

<pre> <presentation label="MQ-Frage"> <flow> <material> <mattext texttype="text/html"><p>Wähle die zugehörigen Paare</p></mattext> </material> <response_grp ident="MQ" rcardinality="Multiple"> <render_choice shuffle="Yes"> <!-- Definition 1 ----- --> <response_label ident="69363" match_max="1" match_group="46339,27242"> <material> <matimage imagtype="image/jpeg" label="682d1150c4.png" embedded="base64"> iVBORw0KGgoAAAANSUgAAAYAAAAEs =</matimage> <mattext texttype="text/plain">Definition 1</mattext> </material> </response_label> <!-- Definition 2 ----- --> <response_label ident="57603" match_max="1" match_group="46339,27242"> <material> <matimage imagtype="image/jpeg" label="1f8bf9512.png" embedded="base64"> iVBORw0KGgoAAAANSUgAAAYAAAAEs =</matimage> <mattext texttype="text/plain">Definition 2</mattext> </material> </response_label> <!-- Term 1 ----- --> <response_label ident="46339"> <material> <matimage imagtype="image/jpeg" label="228a2fd2c74.jpg" embedded="base64"> iVBORw0KGgoAAAANSUgAAAYAAAAEs =</matimage> <mattext texttype="text/plain">Term 1</mattext> </material> </response_label> <!-- Term 2 ----- --> <response_label ident="27242"> <material> <matimage imagtype="image/jpeg" label="8a87d679cf8.gif" embedded="base64"> iVBORw0KGgoAAAANSUgAAAYAAAAEs =</matimage> <mattext texttype="text/plain">Term 2</mattext> </material> </response_label> </render_choice> </response_grp> </flow> </presentation> </pre>	<p>presentation -> label: Hier steht der Fragentitel. Es muss der gleiche Eintrag wie unter „item ident“ sein</p> <p>presentation -> flow -> material -> mattext -> texttype: Gibt die Formatierung des nachfolgenden Textes an. Kann als „fix“ angesehen werden.</p> <p>presentation -> flow -> material -> mattext -> Text: Beschreibt den Fragen-Text (Haupt-Text). Muss zwingend in „<p> </p>“ eingeklammert sein!</p> <p>response_grp -> Ident & rcardinality: „MQ“ und „Multiple“ stehen für eine „Zuordnungsfrage“ Antwort.</p> <p>response_grp -> render_choice -> shuffle: Gibt an ob die Reihenfolge der Definitionen/Terme gemischt werden sollen.</p> <p>response_grp -> ... -> response_label -> ident: Gibt die ID der Definition/ des Terms an. Jede ID darf nur einmalig vorkommen.</p> <p>response_grp -> ... -> response_label -> match_max: Eintrag ist nur für Definitionen relevant. Ist fix auf „1“. Eine Änderung des Wertes bringt keine ersichtliche Änderung im ILIAS.</p> <p>response_grp -> ... -> response_label -> match_group: Eintrag ist nur für Definitionen relevant. Hier sind alle verfügbaren IDs der Terme aufgelistet.</p> <p>response_grp -> ... -> mattext -> texttype: Gibt die Formatierung des nachfolgenden Textes an. Kann als „fix“ angesehen werden.</p> <p>response_grp -> ... -> mattext -> Text: Spiegelt den Antwort-Text wider.</p> <p>response_grp -> ... -> matimage -> imagtype: Für eine Zuordnungsfrage achtet ILIAS (scheinbar) nicht auf das Bildformat. Kann als fix betrachtet werden.</p> <p>response_grp -> ... -> matimage -> label: Gibt den gesamten Namen des Bildes an (mit Format)</p> <p>response_grp -> ... -> matimage -> embedded: Die verwendeten Bilder, innerhalb einer Singlechoice Antwort, sind „Base64“ codiert. Das codierte Bild wird als „String“ dargestellt.</p>
---	--

Tabelle 31: Darstellung der XML-Fragenstruktur für eine Zuordnungsfrage – Definition für Definitionen und Terme – 2/3

Zuordnungsfrage: Definition – Punktevergabe und Rückmeldung für Definition/Term-Paare

<pre> <resprocessing> <!-- ----- Punkte für Definition/Term – Paar 1 ----- --> <respcondition continue="Yes"> <conditionvar> <varsubset respident="MQ">46339,69363</varsubset> </conditionvar> <setvar action="Add">1</setvar> <displayfeedback feedbacktype="Response" linkrefid="correct_46339_" /> </respcondition> <!-- ----- Punkte für Definition/Term – Paar 2 ----- --> <respcondition continue="Yes"> <conditionvar> <varsubset respident="MQ">27242, 57603</varsubset> </conditionvar> <setvar action="Add">1</setvar> <displayfeedback feedbacktype="Response" linkrefid="correct_27242_" /> </respcondition> </resprocessing> <!-- ----- Rückmeldung für Definition/Term Paar 1 ----- --> <itemfeedback ident="correct_46339_69363" view="All"> <flow_mat> <material> <mattext/> </material> </flow_mat> </itemfeedback> <!-- ----- Rückmeldung für Definition/Term Paar 2 ----- --> <itemfeedback ident="correct_27242_57603" view="All"> <flow_mat> <material> <mattext/> </material> </flow_mat> </itemfeedback> </item> </questestinterop> </pre>	<p>respcondition -> continue: Jede Singlechoice Antwort beginnt mit einem „continue=“Yes“ Eintrag. Gibt an ob eine Antwort zur Verfügung steht.</p> <p>varsubset -> respident: „MQ“ steht für eine „Zuordnungsfrage“ Frage</p> <p>varsubset -> Text: Hier stehen die richtigen Zuordnungspaare. Reihenfolge: (Term-ID, Definition-ID)</p> <p>setvar -> action: „Add“ addiert die nachfolgenden Punkte bei richtigen Auswahl der Paare.</p> <p>setvar -> Text: Gibt die Anzahl der Punkte für die Antwort an.</p> <p>displayfeedback -> feedbacktype: Eintrag ist fix auf „Response“ für jede Antwort</p> <p>displayfeedback -> linkrefid: Eintrag fix „correct_“ gefolgt von der Term-ID</p> <p>Itemfeedback -> ident: Eintrag fix „correct_“ gefolgt von der Term ID/ Definition-ID.</p> <p>Itemfeedback -> view: Eintrag ist fix auf „All“ und muss nicht verändert werden.</p>
---	---

Tabelle 32: Darstellung der XML-Fragenstruktur für eine Zuordnungsfrage – Definition Punktevergabe und Rückmeldung für Definition/Term-Paare – 3/3

5 Alternative GUI zu Aufgabenverwaltung und -erstellung

5.1 Anwenderhinweise zur alternativen GUI

Aufbauend auf dem ILIAS-bezogenen Funktionsumfang wurde eine weitere Version des Aufgabengenerators erstellt. Hierbei sollte die Benutzeroberfläche übersichtlicher gestaltet werden und soweit modular aufgebaut werden, dass zukünftige Erweiterungen leichter hinzugefügt werden können. Um diese Software verwenden zu können, muss ebenfalls die Installationsanweisung beachten werden.

5.1.1 Erstmalige Verwendung der Software

Bevor das eigentliche Programm gestartet wird, muss die DB_creator.py ausgeführt werden. Dies wird zwei Datenbanken im Projektverzeichnis erstellen. Dieser Schritt ist nur notwendig, wenn die Datenbank generaldb.db und Zwischenspeicher_Datenbank.db nicht im Projektverzeichnis existieren. Beim ersten Öffnen des Programmes ist die Fragendatenbank leer. Es können jetzt beliebig viele Fragen erstellt und der Datenbank hinzugefügt werden (Button „neue Frage erstellen“).

Nun muss der Fragentyp gewählt werden, welcher erstellt werden soll. Wurden alle Eingabe getätigt kann diese Frage mit dem Button „Als neue Frage speichern“ in der Datenbank gespeichert werden. In der Konsole wird hierzu eine Meldung angezeigt. Bei einer Änderung der Frage kann mit „Änderungen Speichern“ die aktuelle Frage editiert werden.

Mit einem Doppelklick auf den Bildpfad wird ein Preview des hinterlegten Bildes angezeigt.

Achtung:

Fragentitel dürfen nicht nur aus Zahlen bestehen!

Diese Fragen können sonst nicht mehr geöffnet werden.

5.1.2 Frage nachträglich bearbeiten

Fragen werden durch einen Doppelklick auf die entsprechende Zeile geöffnet. Nachdem die entsprechenden Änderungen vorgenommen wurden, kann „Änderungen Speichern“ gedrückt werden.

5.1.3 Test/ Fragenpool zusammenstellen

Zunächst muss eine Auswahl an Fragen zusammengestellt werden. Hierfür linksklicken auf die Fragen die ausgewählt werden sollen. Wenn diese dann blau markiert ist, kann „Frage zu Test hinzufügen“ gedrückt werden. So wird die Frage in die untere Datenbank kopiert. Es ist auch möglich mehrere Fragen gleichzeitig mit „Shift“ oder „Strg“ auszuwählen und dann mit dem „Frage zu Test hinzufügen“ Button zur Auswahl hinzuzufügen.

Über den „Test erstellen“ Button kann ein Test erzeugt werden. Im Kontextmenü muss ein Name eingegeben werden und anschließend besteht die Möglichkeit Testeinstellungen einzubinden. Wenn kein Profil für Testeinstellungen verwendet wird, wird ein Standardprofil verwendet.

Für einen Fragen Pool kann der „Pool erstellen“- Button verwendet werden.

5.1.4 Hilfe

Auf der Startseite finden Sie im rechten Menü einen Hilfebutton, hier finden Sie nochmal die nicht ganz offensichtlichen Funktionen erklärt.

5.2 Software Aufbau der alternativen GUI

Link: <https://github.com/jakobis95/ILIAS---Test-Generator>

Prinzipiell ist die Benutzeroberfläche von den Funktionalitäten getrennt. In der Main.py werden alle benötigten Instanzen initialisiert und die Startseite erstellt. Das Grundprinzip ist, dass alle Klassen, welche Oberflächen darstellen, Abonnenten(Subscriber) der Klassen, die Daten verarbeiten und bereitstellen, sind. Eine zentrale Rolle spielt hier das DB_Interface. Es übernimmt sämtliche Interaktionen mit der Datenbank wie z.B. Suchen, Löschen und Erstellen von Fragen. Diese Funktionen können von den UI Klassen aufgerufen werden. Wird eine Funktion von DB_Interface aufgerufen, sendet DB_Interface ein immer gleich strukturiertes Datenpaket an alle Abonnenten. Jeder Abonnent sucht sich dann den für ihn interessanten Teil aus dem Datenpaket heraus. Dies geschieht in der update()-Funktion, welche in diesem Fall von der DB_Interface-Klasse aufgerufen wird.

5.2.1 Kurze Beschreibung der .py Projektdateien

5.2.1.1 Disclaimer:

Die Abbildungen sind auf MacOS entstanden. Auf Windows und Linux werden die Farben richtig dargestellt. Aus einem nicht bekannten Grund werden bei einem Mac die Buttonfarben nicht angezeigt, daher sind alle Buttons hell. Auf Windows sind diese Buttons dunkel.

5.2.1.2 DB_Interface.py

Hier finden sich alle Funktionen die mit der Kommunikation zwischen Oberfläche und Datenbank zu tun haben. Die verwendete Datenbank ist auf sqlite3 Basis. Alle UI Elemente, die Daten aus einer Datenbank beziehen möchten, müssen sich über die subscribe(self.update()) -Funktion von DB_Interface als neuen Abonnent bekannt machen. Eine weitere Funktion die DB_Interface übernimmt, ist eine Reihe an Listen und Dictionaries mit String-Variablen zu erzeugen. Diese bilden die Tabellen und Spalten der Datenbank ab und werden gleichzeitig als Zwischenspeicher bei der Fragenerstellung verwendet. Mit den String-Variablen spart man sich die Abfrage von Entry-Feldern, da diese direkt als Textvariable verknüpft werden und somit immer denselben Wert, wie das Entry-Feld, haben. Mit den Dictionaries kann der Spaltenname der Datenbank verwendet werden, um den Index der entsprechenden String-Variable zu bestimmen. Im Detail wird dies im nächsten Kapitel gezeigt.

5.2.1.3 Main.py

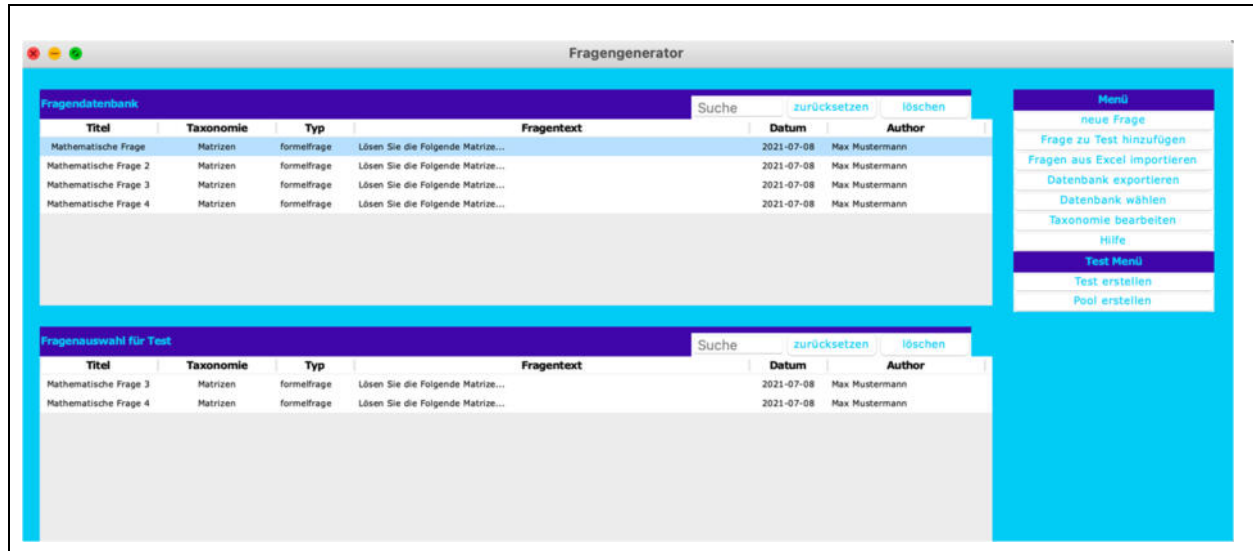


Abbildung 59: Startseite Aufgabengenerator 2.0

Hier wird das Programm gestartet und alle Initialisierungen durchgeführt. Hier befindet sich auch das Hauptmenü und es werden die entsprechenden Instanzen der anderen UI_Elemente aufgerufen. Die beiden Tabellenansichten der Datenbank und des „Fragenauswahl für Test“ wird durch zwei Instanzen der Klasse UI in DB_Treeview.py erstellt. Der Name DB_Treeview kommt daher, dass diese Art der Tabelle unter Tkinter in Python Treeview heißt. Neue Menüpunkte und Funktionen sollten hier eingebunden werden. Jede neue Klasse, die auf die Datenbank zugreifen will, muss diese auch kennen.

5.2.1.4 Test_Klasse.py

In dieser Klasse sind alle Funktionen, die mit dem Erstellen und Verwalten von Testeinstellungsprofilen zu tun haben und der Aufruf der Funktion für die Erstellung des Tests.

5.2.1.5 DB_Treeview.py



Abbildung 60: Datenbank in Treeviewansicht

Wie bereits beschrieben wird hiermit eine Auswahl an Parametern der Fragendatenbank in scrollbarer Tabellenform angezeigt. Das Interface besteht zusätzlich aus einem Aufruf der LösCHFunktion und einem Suchfeld welches mit Enter ausgeführt wird.

5.2.1.6 DB_creator.py

DB_creator beinhaltet eine Klasse, welche leere Datenbanken erzeugen kann. Leere Datenbank bezieht sich lediglich auf die gespeicherten Daten. In dieser Datei wird der Aufbau der Datenbank beschrieben mit den verschiedenen Tabellen und deren Spalten. Änderungen am Datenbanken Aufbau, wie eine weitere Variable für Formelfragen, werden hier durchgeführt. Das Ausführen dieser Datei erzeugt zwei leere Datenbanken. Diese Klasse wird auch verwendet um unter dem Button „Datenbank wählen“ eine neue Datenbank zu erzeugen.

Wichtig:

Dabei müssen die Datenbanken dbgeneral.db und dbgeneral2.db vorher umbenannt werden oder aus dem Projektverzeichnis gelöscht werden.

5.2.1.7 Variablen_Einfügen_UI.py

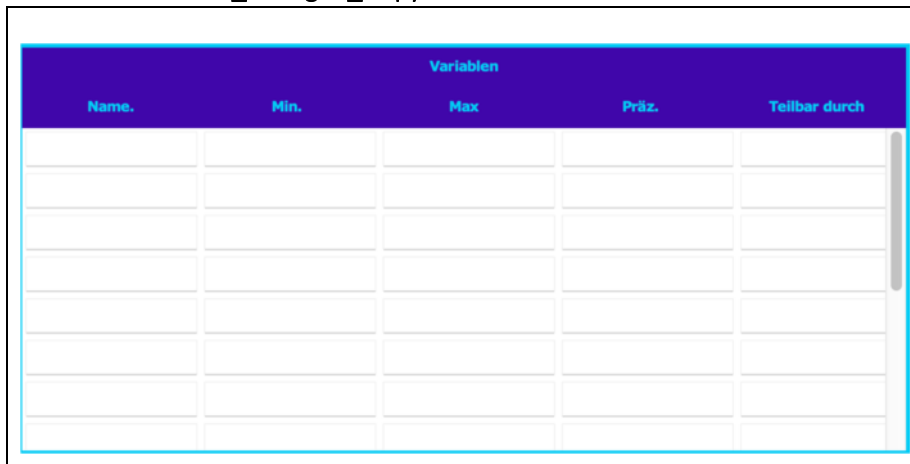


Abbildung 61: Variablen_Einfügen_UI für Formelfragen Variablen implementiert

Jede der Fragentypen hat eine Art von Variablen/ Fragen-Antworten-Teil der aus mehreren Spalten und Reihen besteht. Daher wurde eine Klasse, welche diese UI-Elemente erstellt geschaffen. Die Anzahl an Spalten und Reihen kann beliebig gewählt. Jedes Feld kann eine von 3 Funktionen haben: Entry, Picture-interface oder combobox. Wobei Picture_interface eine Eigenentwicklung ist, welche so nicht in Tkinter oder Python existiert.

5.2.1.8 Picture_interface.py

Picture_interface soll die grafische Eingabe von Bildern ermöglichen. Dabei wird lediglich der Pfad des Bildes gespeichert, weshalb eine eigene Funktion geschrieben wurde. Des Weiteren gehört die Bedienoberfläche (Button/ Label) und die Möglichkeit sich eine Bildvorschau, durch einen Doppelklick auf den Bild-Pfad, anzeigen zu lassen zu den Funktionen.

5.2.1.9 XML_class.py

Hier ist es ähnlich zu dem Funktionskonzept des DB_Interface. Die XML_class beherbergt alle Funktionalitäten/ Klassen welche sich um die Erstellung von XML-Dateien drehen, wie sie schließlich von ILIAS verwendet werden.

5.2.1.10 Time_input_UI.py

Hier findet sich die UI, über welche die Bearbeitungszeit eines Tests eingegeben/angezeigt wird, wieder.

5.2.1.11 Fragen_GUI.py

Abbildung 62: Fragen GUI Layout für Formelfragen

Abbildung 63: Variablen-Interface für Singlechoice-Fragen

Abbildung 64: Variableninterface für Zuordnungsfrage

In Fragen_GUI befindet sich eine Parentklasse für das Erstellen und Bearbeiten von Aufgaben. Wie in Abbildung 62 zu sehen ist, kann der Bildschirm in eine rechte und eine linke Seite aufgeteilt werden. Die linke Seite ist für jeden Aufgabentyp gleich. Auf der rechten Seite ändern sich die Eingabemöglichkeiten entsprechend dem Aufgabentyp. Um für jeden Fragentyp dieses individuelle Variableninterface zu erstellen, gibt es eine child-Klasse für jede Frage in der diese Konfigurationen vorgenommen werden. Nach dem gleichen Schema lässt sich so auch ein neuer Fragentyp hinzufügen.

5.2.2 Code Logik für die Integration neuer Funktionen

An dieser Stelle sollen einige Codefunktionen am Beispiel erklärt werden, um gegebenenfalls selbständige Erweiterungen oder Optimierungen der Software zu ermöglichen

5.2.2.1 Index-Liste und Index-dict (Daten zuordnen)

Bei Start des Programms wird die `DB_interface` Funktion `get_index_info()` ausgeführt. Hierfür muss eine Liste an die Instanz von `DB_interface` übergeben worden sein, in der alle verwendeten Tablenamen gespeichert sind. Dann werden zwei leere Listen erstellt. Erstens `table_index_list`, hier werden die für jeden Table `index_list` gespeichert. Zweitens die `table_index_dict` (dictionary). In diesem Dictionary wird für jeden Table ein `index_dict` gespeichert.

```
def get_index_info(self):  
    # hier sind die index_list elemente für jeden table die den Fragentypenentsprechen zusammengefasst  
    self.table_index_list = [None, None, None, None, None, None]  
  
    # hier sind die index_dict elemente für jeden table zusammengefasst  
    self.table_index_dict = [None, None, None, None, None, None]
```

Abbildung 65: Erstellung `table_index_list` und `table_index_dict`

Jetzt werden für jeden `table` aus der `table_list` eine leere `index_list` und `index_dict` erstellt.

```
for table in self.table_list:  
    self.index_list = []  
    self.index_dict = {}  
    index = 0
```

Abbildung 66: Erstellung `index_list` und `index_dict`

Mit dem untenstehenden `sqlite3`-Befehl werden die Table-Reihennamen ausgelesen. Für jede Reihe (`Row`) wird der Eintrag für `index_list` (`q`) und `index_dict` (`d`) erstellt. `q` besteht aus der Stringvariable und dem Spaltennamen als String. `d` besteht aus dem Spaltennamen und dem numerischen Index in diesem `table`.

```
for row in self.cursor:  
    Var = StringVar()  
    q = (Var, row[1])  
    d = {row[1] : index}  
  
    self.index_dict.update(d)  
    self.index_list.append(q)  
    index = index + 1  
  
# liste von dictionary für jeden tabel kann mit self.table_dict[tablename] verwendet werden  
self.table_index_dict[i] = self.index_dict  
self.table_index_list[i] = self.index_list  
i = i + 1
```

Abbildung 67: Auslesender Table-Reihennamen

Mit diesem SQL wird für jede Frage in der Datenbank der Inhalt für diese 6 Spalten ausgegeben. Table_index_list[table_index][row_index][var_index] hat 3 Dimensionen. Um den richtigen Index herauszufinden, kann jetzt jeweils das dict verwendet werden, um den Index zu bestimmen.

`self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["question_title"]][1]`. Die 1 in der letzten Klammer gibt an das der Name der Spalte verwendet werden soll. Eine 0 würde auf die Stringvariable verweisen.

Auf der nächsten Seite findet man den vollständigen Code

```
self.query = "SELECT " + self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["question_title"]][1] + ", " \
+ self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["question_pool_tag"]][1] + ", " \
+ self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["question_type"]][1] + ", " \
+ self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["question_description_main"]][1] + ", " \
+ self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["date"]][1] + ", " \
+ self.table_index_list[self.table_dict[table]][self.table_index_dict[self.table_dict[table]]["question_author"]][1] + " FROM " + table + ""
```

Abbildung 68: Beispiel für Fragen erstellen und verwenden von String-Variablen

Eine Frage besitzt einen Zwischenspeicher der nur die `index_list` für einen `table` ist. Jeder Fragentyp hat seinen eigenen Table. Wird eine Frage zur Bearbeitung geöffnet, werden alle Daten der Frage in diese `index_list` kopiert. Über diese `index_list` kann dann per Name der Spalte die Textvariable zugeordnet werden. Dies sieht man in der folgenden Abbildung. Beim Erstellen der `Fragen_GUI` Instanz wird das `index_dict` für den entsprechenden Fragentyp/`table` mitgegeben. Ändert sich jetzt die Reihenfolge der Spalten in der Datenbank führt dies nicht zu Fehlern im Programm.

```
self.PoolTag_Entry = Entry(self.param_Frame,
                           textvariable=self.dbinhaltsliste[self.index_dict['question_pool_tag']][0],
                           bg=self.entry_color, fg=self.efg_color)
```

Abbildung 69: Beispiel für Übergabe in Dictionary

6 Sonstiges

6.1 Ausführbare Datei aus Programmcode erstellen

Um aus einem vorhandenen Projekt eine ausführbare Datei zu erstellen, soll zunächst eine Virtuelle Umgebung über Anaconda eingerichtet werden. Diese Virtuelle Umgebung wird dazu genutzt das Projekt mit nur essentiellen Bibliotheken zu starten. Dadurch kann die Größe der Ausführbaren Datei stark reduziert werden, da der Befehl zur Konvertierung vom Projekt zur .exe sämtliche installierten Bibliotheken miteinbezieht.

6.1.1 Virtual-Environment aufsetzen

Zunächst wird das „Anaconda Prompt“ als Administrator ausgeführt. Nun wird eine Konsole dargestellt, in welcher Befehle eingegeben werden können.

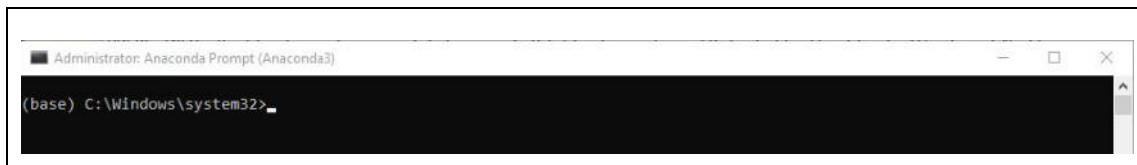


Abbildung 70: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 1/4

Anschließend wird mit dem Befehl „cd“ (change directory), zum Projekt gewechselt.

Dann kann durch den Befehl: „conda create -n „Name für Umgebung“ python=3.8“ eine neue virtuelle Umgebung angelegt werden.

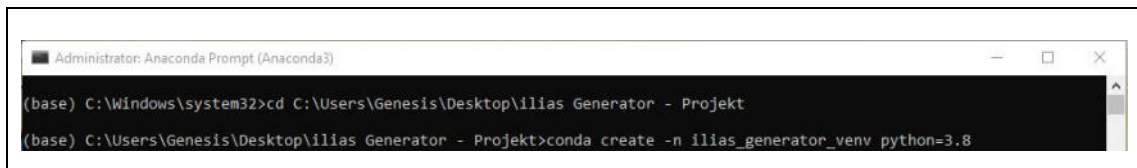


Abbildung 71: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 2/4

Nach dem Ausführen des Befehls sollen einige Grundpakete installiert werden. Hier mit einem „y“ bestätigen.

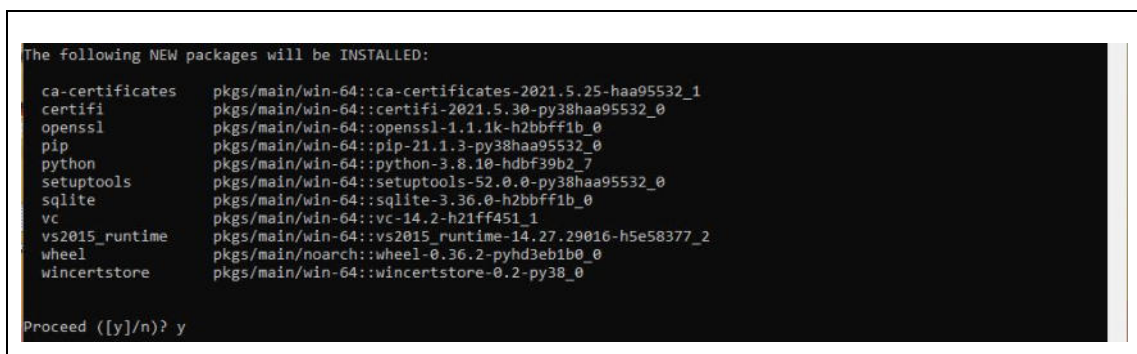


Abbildung 72: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 3/4

Nach erfolgreicher Erstellung wird über „activate „Name der Umgebung““ die virtuelle Umgebung aktiviert:

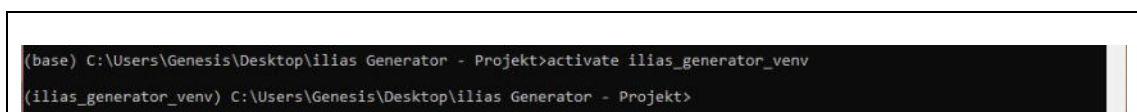


Abbildung 73: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 4/4

6.1.2 Notwendige Bibliotheken installieren und ausführbare Datei erstellen

Um alle derzeitig installierten Bibliotheken angezeigt zu bekommen, kann der Befehl „conda list“ verwendet werden:

```
(ilias_generator_venv) C:\Users\Genesis\Desktop\ilias Generator - Projekt>conda list
# packages in environment at C:\ProgramData\Anaconda3\envs\ilias_generator_venv:
#
# Name                    Version            Build    Channel
ca-certificates           2021.5.25          haa95532_1
certifi                   2021.5.30          py38haa95532_0
openssl                   1.1.1k             h2bbff1b_0
pip                       21.1.3             py38haa95532_0
python                    3.8.10             hdbf39b2_7
setuptools                52.0.0             py38haa95532_0
sqlite                    3.36.0             h2bbff1b_0
vc                         14.2               h21ff451_1
vs2015_runtime            14.27.29016        h5e58377_2
wheel                     0.36.2             pyhd3eb1b0_0
winertstore                0.2                py38_0

(ilias_generator_venv) C:\Users\Genesis\Desktop\ilias Generator - Projekt>
```

Abbildung 74: Ausführbare Datei aus Programmcode erstellen – Auflistung bereits installierter Bibliotheken

Für die Erstellung der .exe wird hier die Bibliothek „pyinstaller“ verwendet und muss separat installiert werden „pip install pyinstaller“.

Über den Befehl „pyinstaller --onefile „Dateiname.py“ wird aus dem Projekt eine .exe erstellt. Der Zusatz „--onefile“ dient dazu nur eine einzige Datei zu erstellen und nicht noch zusätzliche, nicht notwendige, Dateien.

Es werden sämtliche Abhängigkeiten (von anderen .py Dateien) mitberücksichtigt. Die Dauer zur Erstellung der .exe kann (je nach Größe) stark variieren.

```
(ilias_generator_venv) C:\Users\Genesis\Desktop\ilias Generator - Projekt>pyinstaller --onefile ILIAS_Test_Generator.py
```

Abbildung 75: Ausführbare Datei aus Programmcode erstellen – Befehl zur Erstellung einer .exe

Nach Abschluss der Erstellung liegt im Projekt-Ordner ein neuer Ordner „dist“. In diesem Ordner befindet sich die neu erstellte .exe. Um diese richtig verwenden zu können, muss die Datei im Pfad verschoben werden. Da die alten Abhängigkeiten noch gelten, ist der Pfad „ilias Generator - Projekt\dist*.exe“ für das Projekt nicht bekannt. Die .exe muss, in diesem Beispiel, in das übergeordnete Verzeichnis „ilias Generator – Projekt“ verschoben werden. Unter diesem Pfad lag auch die ILIAS_Test_Generator.py.

Beim Versuch diese .exe auszuführen, wird eine Fehlermeldung über fehlende Bibliotheken angezeigt. In der neu erstellten „ilias_generator_venv“ sind noch keine Bibliotheken installiert und daher auch nicht für das Programm verfügbar.

```
(ilias_generator_venv) C:\Users\Genesis\Desktop\ilias Generator - Projekt>ILIAS_Test_Generator.exe
Traceback (most recent call last):
  File "ILIAS_Test_Generator.py", line 73, in <module>
    from tkscrolledframe import ScrolledFrame #Bewegbares Fenster (Scrollbalken)
ModuleNotFoundError: No module named 'tkscrolledframe'
[21428] Failed to execute script ILIAS_Test_Generator

(ilias_generator_venv) C:\Users\Genesis\Desktop\ilias Generator - Projekt>
```

Abbildung 76: Ausführbare Datei aus Programmcode erstellen – Fehlermeldung wegen fehlender Bibliotheken

Über die manuelle Installation der benötigten Pakete wird schließlich die Größe der .exe reduziert, da nur die essentiellen Pakete „aktiv“ sind. Kontrolle der installierten Bibliotheken über den Befehl „conda list“.

Name	Version	Build	Channel
altgraph	0.17	pypi_0	pypi
blas	2.108	openblas	conda-forge
blas-devel	3.9.0	8 openblas	conda-forge
ca-certificates	2020.12.5	h5b45452_0	conda-forge
certifi	2020.12.5	py38haa244fe_1	conda-forge
click	7.1.2	pypi_0	pypi
cycler	0.10.0	pypi_0	pypi
et-xmlfile	1.0.1	pypi_0	pypi
future	0.18.2	pypi_0	pypi
hissolver	1.3.1	pypi_0	pypi
libblas	3.9.0	8 openblas	conda-forge
libblas	3.9.0	8 openblas	conda-forge
libflang	5.0.0	h6538335_20180525	conda-forge
liblapack	3.9.0	8 openblas	conda-forge
liblapack	3.9.0	8 openblas	conda-forge
libopenblas	0.3.12	pthread_h1662909_1	conda-forge
llvm-meta	5.0.0	0	conda-forge
lz4	5.3.0	6	conda-forge
lz4-gcc-libgfortran	5.3.0	7	conda-forge
lz4-gcc-libstdc++	5.3.0	7	conda-forge
lz4-gcc-libstdc++-core	5.3.0	7	conda-forge
lz4-gmp	6.1.0	2	conda-forge
lz4-libwinpthread-git	5.0.0.4634.697f757	2	conda-forge
mpmath	1.2.1	pypi_0	pypi
msys2-conda-epoch	20160418	1	conda-forge
numpy	1.20.2	py38h09042cb_0	conda-forge
openblas	0.3.12	pthread_h6ebc607_1	conda-forge
openmp	5.0.0	vc14_1	conda-forge
openpyxl	3.0.7	pypi_0	pypi
openssl	1.1.1k	h8ffe710_0	conda-forge
pandas	1.2.3	pypi_0	pypi
perfile	2019.4.18	pypi_0	pypi
pillow	8.2.0	pypi_0	pypi
pip	21.0.1	py38haa95532_0	pypi
rmu	2.0.1	pypi_0	pypi

Abbildung 77: Ausführbare Datei aus Programmcode erstellen – Auflistung aller verwendeten Bibliotheken 1/2

pyinstaller	4.2	pypi_0	pypi
pyinstaller-hooks-contrib	2021.1	pypi_0	pypi
pyarsing	2.4.7	pypi_0	pypi
pywin32	223	pypi_0	pypi
pyqt5	5.15.4	pypi_0	pypi
pyqt5-plugins	5.15.4.2.1.0	pypi_0	pypi
pyqt5-qt5	5.15.2	pypi_0	pypi
pyqt5-sip	12.9.0	pypi_0	pypi
pyqt5-tools	5.15.4.3.0.3	pypi_0	pypi
python	3.8.8	hdbf39b2_4	pypi
python-dateutil	2.8.1	pypi_0	pypi
python-dotenv	0.17.1	pypi_0	pypi
python_abi	3.8	1_cp38	conda-forge
pytz	2021.1	pypi_0	pypi
pywin32	300	pypi_0	pypi
pywin32-ctypes	0.2.0	pypi_0	pypi
qt5-applications	5.15.2.2.1	pypi_0	pypi
qt5-tools	5.15.2.1.0.1	pypi_0	pypi
setuptools	52.0.0	py38haa95532_0	pypi
six	1.15.0	pypi_0	pypi
sqlite	3.35.4	h2bbff1b_0	pypi
sympy	1.7.1	pypi_0	pypi
tkscrolledframe	1.0.4	pypi_0	pypi
vc	14.2	h21ff451_1	pypi
vs2015_runtime	14.27.29016	h5e58377_2	pypi
wheel	0.36.2	pyhd3eb1b0_0	pypi
wincertstore	0.2	py38_0	pypi
xlsxwriter	1.3.8	pypi_0	pypi

Abbildung 78: Ausführbare Datei aus Programmcode erstellen – Auflistung aller verwendeten Bibliotheken 2/2

Mit Hilfe des Pyinstaller wird nun eine .exe generiert. Pyinstaller hat das Problem, dass manche Bibliotheken nicht erkannt werden. Um alle notwendigen Bibliotheken verwenden zu können kommt der Befehl „--hidden-import“ zum Einsatz. Hierdurch werden explizit die Bibliotheken genannt die verwendet werden sollen. In diesem Beispiel sieht die Kommandozeile wie folgt aus:

```
pyinstaller
--hidden-import tkscrolledframe
--hidden-import xlsxwriter
--hidden-import pandas._libs.tslibs.base
--hidden-import sympy
--hidden-import PIL
--hidden-import matplotlib
--onefile ILIAS_Test_Generator.py
```

Zur Übersichtlichkeit wurden Zeilenumbrüche hinzugefügt. In der Kommandozeile werden die Befehle jeweils mit einem Leerzeichen getrennt eingegeben:
(pyinstaller --hidden-import tkscrolledframe --hidden-import xlsxwriter ...)

Die bereits erstellte virtuelle Umgebung kann aus einem anderen Repository bezogen werden. Die gepackte Größe beträgt etwa 200MB ist damit zu groß für ein normales Git repository.

Link: <https://github.com/TPanteleit/ILIAS---Test-Generator---venv>

Die Verwaltung von virtuellen Umgebungen für Anaconda, kann unter dem Pfad „C:\Users\xxx\conda\envs“ gefunden werden. Hier muss der Ordner schließlich entpackt werden (Größe etwa 700MB!).

TEIL B: Softwaredokumentation für ILIAS-Aufgabengenerator

1 Python Aufbau des Aufgabengenerators

1.1 Programmierung der GUI

Die GUI wurde mit der Python Bibliothek „Tkinter“ programmiert. In jedem zugehörigen Modul des Aufgabengenerators befindet sich in der INIT der entsprechende GUI Teil (Definierung der Buttons etc.). Aus dem „Hauptprogramm“ (ILIAS_Test_Generator.py) werden die verschiedenen Module aufgerufen und somit die GUI erstellt.

1.2 Auflistung der Module und der zugehörigen Funktionen

Der Aufgabengenerator basiert auf verschiedenen Python-Modulen. Das Programm wird über die Datei „ILIAS_Test_Generator.py“ im Projektverzeichnis gestartet. Diese lädt die eigentlichen Module aus „Projektverzeichnis\Test_Generator_Module\“ zur Verwendung der Funktionen.

Hauptdatei zum Start des Aufgabengenerators

- ILIAS_Test_Generator.py

Module für Fragentypen

Diese Module erzeugen die entsprechenden Anzeigen auf der GUI und erstellen die jeweiligen Fragen. In den Modulen ist die XML-Struktur der jeweiligen Frage abgelegt.

- test_generator_modul_formelfrage
- test_generator_modul_singlechoice
- test_generator_modul_multiplechoice
- test_generator_modul_zuordnungsfrage
- test_generator_modul_formelfrage_permutation

Module für zur Erstellung von Test/Pool

Dieses Modul dient zur Erstellung eines Fragentests bzw. –pools. Der XML Aufbau ist jeweils leicht anders (z.B. Testeinstellungen sind für einen Pool nicht vorgesehen).

- test_generator_modul_ilias_test_struktur

Modul für Testeinstellungen

Definiert die Testeinstellungen für einen Fragentest. Es sind nicht alle Einstellungsmöglichkeiten aus dem ILIAS vorhanden, sondern lediglich die Einstellungsmöglichkeiten unter „Allgemeine Einstellungen“.

- test_generator_modul_testeinstellungen

Module für Datenbanken

„Datenbanken anzeigen“ ermöglicht das Auslesen der DB und der Anzeige der jeweiligen Frage im GUI Fenster „DB anzeigen“. „Datenbank erstellen“ beinhaltet auch die Funktionen für einen Import aus einer Tabellenkalkulationsdatei und den Export aus der DB nach z.B. Excel

- test_generator_modul_datenbanken_anzeigen
- test_generator_modul_datenbanken_erstellen

Modul für Import von Fragen aus Test/Pool

Dieses Modul ermöglicht das Auslesen bestehender Fragentests und –pools. Die Fragen werden in die jeweilige DB integriert

- test_generator_modul_ilias_import_test_datei

Module für Taxonomien

Bietet die Funktion erstellte Fragenpools in der Taxonomie zu bearbeiten.

- test_generator_modul_taxonomie_und_textformatierung

Modul für Zeigerdiagramme

Dient dazu aus vordefinierten Schaltungen entsprechende Zeigerdiagramme zu erstellen. Diese Funktion ist für „ILIAS_Test_Generator.exe“ auskommentiert, da die Datei sonst deutlich größer wird. Das Modul „Zeigerdiagramm“ ist nicht stark ausgebaut und bietet lediglich die Option Strom-, Spannungs- und Leistungsdiagramm für folgende Testschaltungen (RC-, RL-, RLC-Schaltung):

- test_generator_modul_zeigerdiagramme

1.2.1 Module für Fragentypen

Nachfolgend werden die verfügbaren Klassen und Funktionen der jeweiligen Module aufgelistet. Die Auflistung dient der Orientierung, eine zugehörige Beschreibung ist im Programmcode zu finden.

1.2.1.1 Formelfrage

Klasse: Formelfrage

- __init__
- ff_variable_show_or_remove
- ff_result_show_or_remove
- unit_table
- ff_replace_character_in_xml_file
- ff_calculate_value_range_function_in_GUI
- ff_calculate_value_range_replace_formula_numpy
- ff_calculate_value_range_from_formula_in_GUI
- ff_save_id_to_db
- ff_load_id_from_db
- ff_edit_id_from_db
- ff_delete_id_from_db
- ff_clear_var_res_entries
- ff_clear_GUI

Klasse: Create_Formelfrage_Questions

- __init__
- ff_question_structure
- ff_question_variable_structure
- ff_question_results_structure

Klasse: Create_Formelfrage_Test

- __init__

Klasse: Create_Formelfrage_Pool

- __init__

1.2.1.2 Formelfrage_Permutation

Klasse: Formelfrage_Permutation

```
__init__  
ffperm_variable_show_or_remove  
ffperm_result_show_or_remove  
unit_table  
ffperm_replace_symbols_in_formula  
ffperm_replace_character_in_xml_file  
ffperm_calculate_value_range_from_formula_in_GUI  
ffperm_save_id_from_db  
ffperm_load_id_from_db  
ffperm_edit_id_from_db  
ffperm_delete_id_from_db  
ffperm_clear_GUI
```

Klasse: Create_Formelfrage_Permutation_Questions

```
__init__  
ffperm_question_structure  
ffperm_question_variable_structure  
ffperm_question_results_structure
```

Klasse: Create_Formelfrage_Permutation_Test

```
__init__
```

Klasse: Create_Formelfrage_Permutation_Pool

```
__init__
```

1.2.1.3 Singlechoice

Klasse: SingleChoice

```
__init__  
sc_add_image_to_answer  
sc_save_id_to_db  
sc_load_id_from_db  
sc_edit_id_from_db  
sc_delete_id_from_db  
sc_clear_GUI
```

Klasse: Create SingleChoice Questions

```
__init__  
sc_question_structure  
sc_question_answer_structure
```

Klasse: Create SingleChoice Test

```
__init__
```

Klasse: Create SingleChoice Pool

```
__init__
```

1.2.1.4 Multiplechoice

Klasse: MultipleChoice

```
__init__  
mc_add_image_to_answer  
mc_save_id_to_db  
mc_load_id_from_db  
mc_edit_id_from_db  
mc_delete_id_from_db  
mc_clear_GUI
```

Klasse: Create MultiplChoice Questions

```
__init__  
mc_question_structure  
mc_question_answer_structure
```

Klasse: Create MultiplChoice Test

```
__init__
```

Klasse: Create MultiplChoice Pool

```
__init__
```

1.2.1.5 Zuordnungsfrage

Klasse: Zuordnungsfrage

```
__init__  
mq_add_image_to_answer  
mq_save_id_to_db  
mq_load_id_from_db  
mq_edit_id_from_db  
mq_delete_id_from_db  
mq_clear_GUI
```

Klasse: Create Zuordnungsfrage Questions

```
__init__  
mq_question_structure  
mq_question_answer_structure_definitions  
mq_question_answer_structure_terms  
mq_question_answer_structure_assignment_pairs
```

Klasse: Create Zuordnungsfrage Test

```
__init__
```

Klasse: Create Zuordnungsfrage Pool

```
__init__
```


1.2.2 Modul zur Erstellung eines ILIAS Test/Pool

Klasse: Create_ILIAS_Test

__init__
test_structure

Klasse: Create_ILIAS_Pool

__init__
pool_structure

Klasse: Additional_Funtions

__init__
Add_picture_to_description_main
Set_picture_in_main
Add_dir_for_images
Replace_character_in_xml_file
Copytree
Set_taxonomy_for_question
Taxonomy_file_refresh
createFolder
add_image_to_description
delete_image_from_description

1.2.3 Modul für Testeinstellungen

Klasse: Test_Einstellungen

__init__
show_entry_time_limited_start
show_introduction_textfield
enable_autosave
show_concluding_remarks
profile_show_db
profile_save_settings
profile_load_settings
profile_delete
profile_delete_last
create_settings

1.2.4 Module für Datenbanken

1.2.4.1 Datenbanken anzeigen

Klasse: MainGUI

__init__

1.2.4.2 Datenbanken erstellen und Import/Export (von/nach Tabellenkalkulationsdatei)

Klasse: CreateDatabases

__init__

create_database_formelfrage

create_database_formelfrage_permutation

create_database_singlechoice

create_database_multiplechoice

create_database_zuordnungsfrage

create_database_test_settings_profiles

Klasse: Import Export Database

__init__

show_duplicates_from_excel_import

excel_import_to_db

excel_import_placeholder_to_db

excel_export_to_xlsx

Klasse: Delete Entry from Database

__init__

1.2.4.3 Import bestehender ILIAS Test/Pool Dateien

Klasse: Import ILIAS Datei in DB

__init__

read_description_main_text_and_img_from_qti

copy_description_main_img_to_dir

copy_response_text_img_to_dir

insert_into_ff_db

insert_into_sc_db

insert_into_mc_db

insert_into_mq_db

find_max_oid_in_table

1.2.5 Modul für Taxonomie/Textformatierung

Klasse: Textformatierung

__init__
text_latex
text_sub
text_sup
text_italic
reallocate_text
format_description_text_in_xml
set_position_for_picture_1
set_position_for_picture_2
set_position_for_picture_3

Klasse: Taxonomie

__init__
select_xml_file_to_read
read_taxonomy_file
update_taxonomy_name
tax_file_refresh
add_node_to_tax
add_node_to_tax_from_excel
assign_questons_to_node_from_excel
remove_node_from_tax
tax_reallocate
tax_reallocate_from_excel
tax_combobox_refresh
scan_tax_tree
assign_questions_to_node
create_taxonomy_for_pool
remove_question_from_node

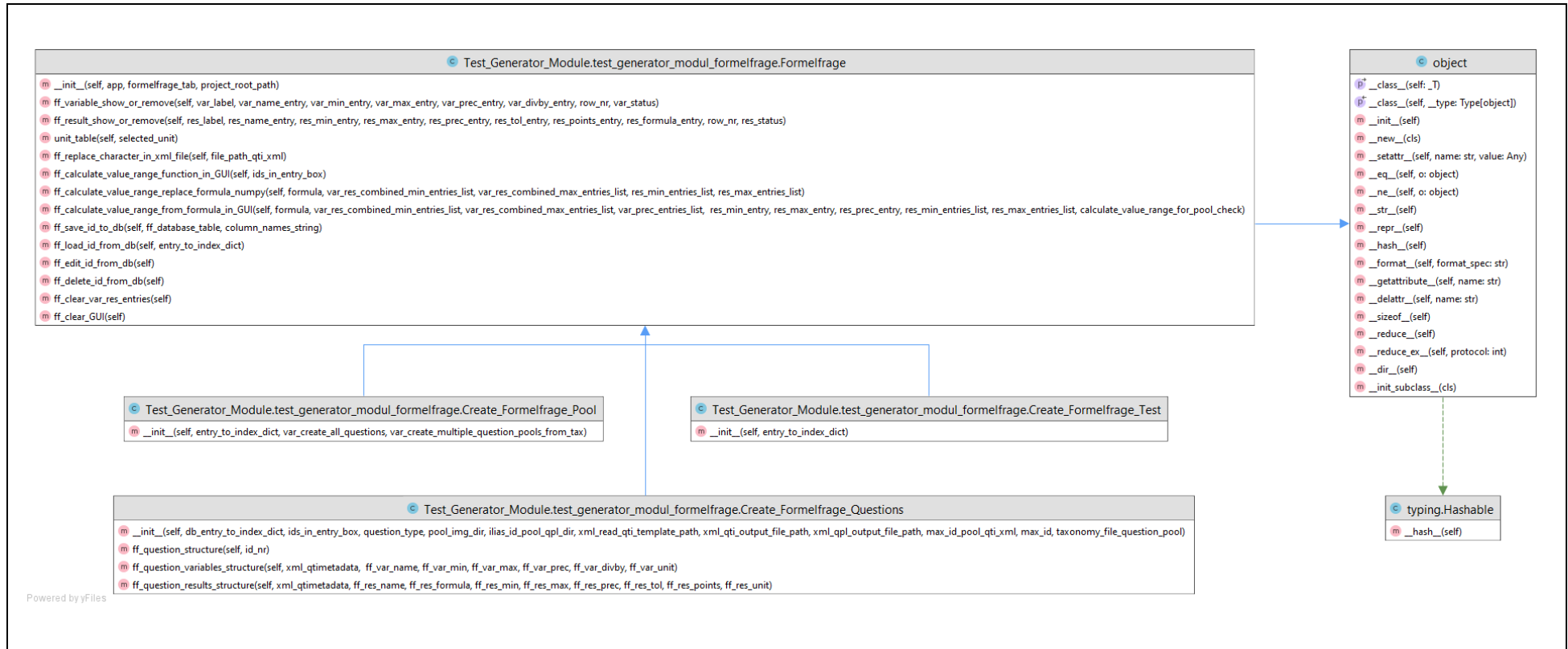
1.2.6 Modul für Zeigerdiagramme

Klasse: Zeigerdiagramme

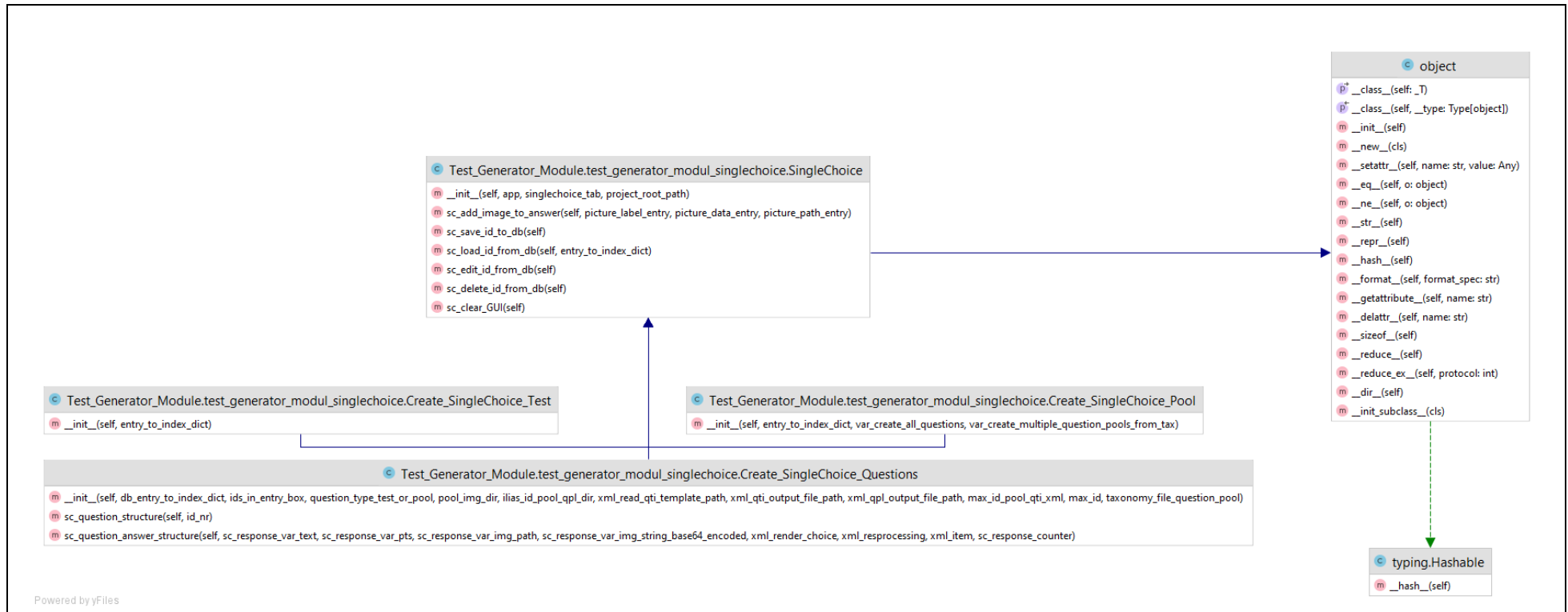
__init__
serienschaltung_R_L
serienschaltung_R_C
serienschaltung_R_L_C
zeigerdiagramm_erstellen

1.3 UML-Diagramme für Fragentyp-Module

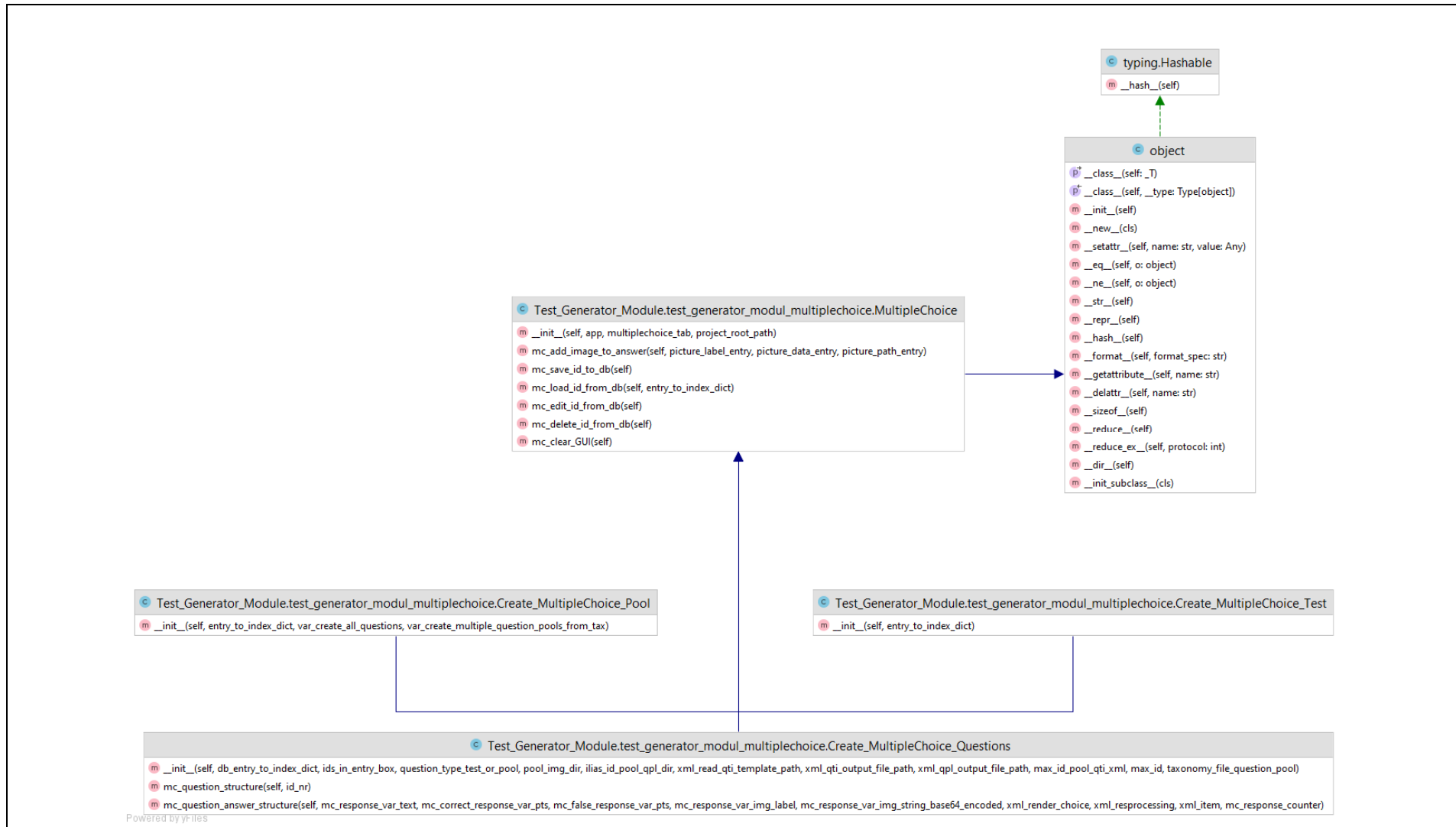
1.3.1 Fragentyp: Formelfrage



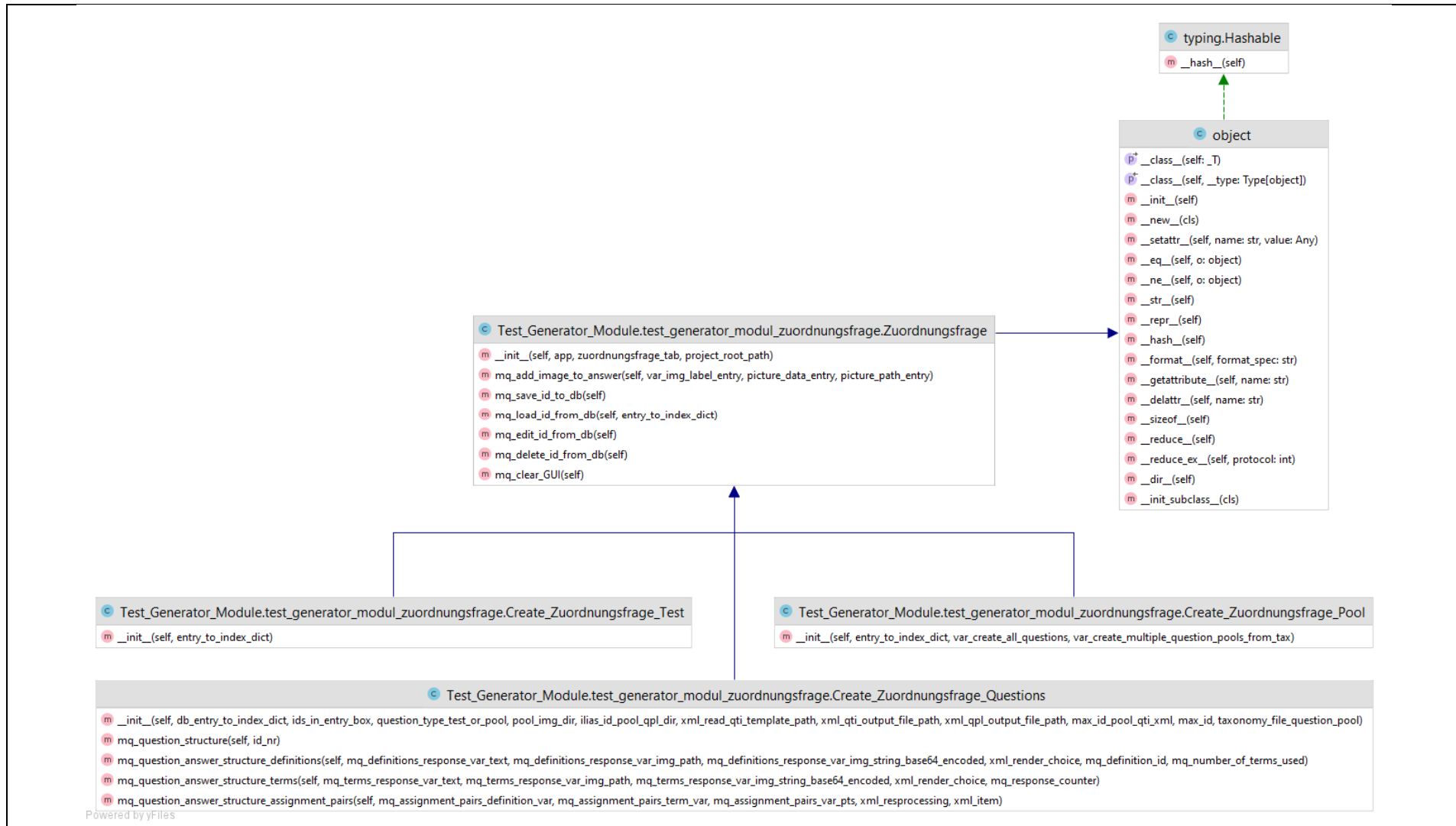
1.3.2 Fragentyp: Singlechoice



1.3.3 Fragentyp: Multiplechoice



1.3.4 Fragentyp: Zuordnungsfrage



Teil C: Auswerte- und Analysetool

1 Nachkorrektur und Auswertung

Das Analysetool kann unter folgendem Link bezogen werden:

Link: <https://github.com/TPanteleit/ILIAS---Test-Generator>

1.4 Beschreibung des Nachkorrekturtools

Das Nachkorrekturtool dient zum Einlesen und Weiterverarbeiten von ILIAS Testergebnissen. Es liest die Ergebnisse eines ILIAS-Formelfragentest, sowie den dazugehörigen Fragenpool aus einer EXCEL-Tabelle ein. Die beiden Tabellen werden miteinander verknüpft und das Tool führt eine elektronische Korrektur des ILIAS-Tests, anhand der im EXCEL-Fragenpool hinterlegten Formel durch. Eventuell gesammelte Bonuspunkte im Semester, bspw. durch Praktika oder Zwischentests können über eine weitere, optionale Tabelle mit Bonuspunkten eingelesen werden. Diese Daten werden über die Matrikelnummer mit den Teilnehmern der Prüfung verknüpft und eine Endwertung des Studierenden anhand eines Notenschemas ermittelt. Die Matrikelnummern werden direkt aus der PSSO-Anmelde-Liste eingelesen. Nicht angemeldete Teilnehmer werden in der Konsole angezeigt und haben in der Ergebnisliste keine Matrikelnummer.

Die vorliegende Version ist folgendermaßen limitiert:

- Auswertung von ausschließlich Formelfragen und Freitextaufgabe
- Single- und Multiple-Choice und andere Fragetypen werden nicht unterstützt. Diese werden ignoriert
- bei Freitextaufgaben können nur Fragen berücksichtigt werden, bei denen ein einzelnes Stichwort im Freitext enthalten sein muss. Das Stichwort für Freitextaufgaben muss im EXCEL_Fragenpool in der Spalte für die Formel des ersten Ergebnisses stehen ("res1_formula").

Wichtige Parameter können (und müssen) als Konstanten direkt am Anfang des Codes festgelegt werden. Dazu gehören:

- Anzahl Fragen,
- Anzahl Variablen pro Frage,
- Anzahl möglicher Ergebnisse pro Frage
- Max. Punktzahl im Test
- Notenschema
- Dateinamen der Import-Dateien
- Dateinamen der Excel und PSSO Export Dateien

ILIAS-Ergebnis-Datei:

Diese Datei wird direkt aus ILIAS exportiert und beinhaltet alle Ergebnisse und Antworten der Studierenden. Der Titel der Datei muss bei den Konstanten zu Beginn des Codes definiert werden. Der Standardname lautet "ILIAS_Testergebnisse.xlsx". Die Datei (wie alle anderen auch) muss im selben Verzeichnis wie das Python-Skript liegen.

Die Datei kann in ILIAS mit den folgenden Schritten erzeugt und gedownloadet werden:

- Statistik] -> "Evaluationsdaten exportieren als" "Microsoft Excel"
- [Export]

Der Export kann dabei eine längere Zeit in Anspruch nehmen. Das vorliegende Tool benutzt dabei das Excel-Tabellenblatt "Auswertung für alle Benutzer". Bei einem kleineren Test mit nur wenigen Benutzern, exportiert ILIAS diese Seite jedoch nicht, sondern generiert für jeden Teilnehmer eine eigene, benannte Seite in dem Excel-Dokument. In dem erweiterten Toolkit dieses Projekts findet sich jedoch ein Python-Skript, welches eine solche Datei in das passende Format umwandeln kann.

Idealerweise soll das Datenformat in der exportierten EXCEL-Tabelle folgendermaßen aussehen:

Ergebnisse von Testdurchlauf 1 für Max, Mustermann |

Formelfrage	26.1.1 Blindwiderstand einer Induktivität
\$v1	83.5
\$v2	46.1
\$r1	132.916
Formelfrage	09.1.1 maximale Leistung von Quelle
\$v1	60
\$v2	24
\$r1	1440
:	
usw.	
:	

Ergebnisse von Testdurchlauf 1 für Maria, Musterfrau |

Formelfrage	18.1.1 Induzierte Spannung
\$v1	340
\$v2	3
\$v3	35
\$v4	60
Freitext Eingabe	50.1.01 Freitextfrage für ETAT
Ergebnis	Hier findet sich der Freitext vom Studenten

Der Name steht also immer in der ersten Spalte nach dem Schlüsseltext "Ergebnisse von Testdurchlauf 1 für ..." Dann folgt eine Leerzeile (wird im Tool nicht ausgewertet). Dann folgen die einzelnen Fragen. Eine Formelfrage fängt mit dem Schlüsseltext "Formelfrage" in der ersten Spalte an. In der zweiten Spalte steht dann der Titel der Frage.

Das vorliegende Tool nimmt an, dass der Text bis zum ersten Leerzeichen der ID der Frage entspricht und extrahiert diesen zusätzlich zum Titel.

Nach dem Titel folgen die Variablen mit den für den Teilnehmer generierten Variablenwerten. Es werden nur die Variablen aufgelistet, die auch in der Frage verwendet wurden. Der ILIAS-Name (\$v1 usw.) steht in der ersten Spalte, der dazugehörige Wert in der zweiten. ACHTUNG: Wenn der Teilnehmer die Frage gar nicht geöffnet hat, werden keine Werte generiert und hier nicht aufgelistet. Dann werden die vom Teilnehmer berechneten Werte \$r1 usw. angezeigt. Auch hier taucht \$r1 nur auf, wenn der Teilnehmer auch eine Eingabe gemacht hat.

Fragenpool-EXCEL-Tabelle

Der Titel der Excel-Datei, in dem sämtliche Fragen gespeichert sind, wird ebenfalls zu Beginn bei den Konstanten eingegeben. Der Standardname der Datei lautet "ILIAS_Fragenpool.xlsx". Die Datei muss im selben Verzeichnis wie das Python-Skript liegen. Die Fragenpool-Tabelle hat ein eigenes Format. Jede Zeile entspricht einer Frage im Fragenpool. Die Tabelle muss die passenden Spaltenüberschriften haben. Diese müssen in der 7. Zeile stehen, denn die ersten 6 Zeilen werden übersprungen. Folgende Überschriften müssen mit den dazugehörigen Spalten vorhanden sein. Dabei ist die Reihenfolge der Spalten egal, es können auch noch andere dazwischen sein.

Beispiel:

: 6x Leerzeile bzw. Informationen über Autor, Datum etc.

Question Type	Question Title	res1_formula	res1 tol	res1 pts
Formelfrage	02.2.3 Ohm'sches Gesetz	\$v1/\$v2	5	1
Formelfrage	03.1.1 Leistung	\$v1*\$v2	5	1
Freitextfrage	50.1.1 Farbe des Himmels	blau		3

:

Question Type: Beschreibt die Art der Frage. Kann entweder "Formelfrage" oder "Freitextfrage" haben.

Question Title: Ist der Titel der Frage. Dieses Tool nimmt an, dass der Text bis zum ersten Leerzeichen der ID der Frage entspricht und extrahiert diesen zusätzlich zum Titel. Tool matcht Fragen aus der ILIAS-Ergebnisdatei mit dem Fragenpool dieser Fragen-ID.

res1_formula: Enthält die Formel zur Musterlösung im ILIAS Format. Nur die Formel zu Antwort 1 wird ausgewertet. Bei Freitextaufgaben steht hier das Schlüsselwort, das in der Antwort enthalten sein muss

res1 tol: Enthält die Toleranz des Ergebnisses in %. Hier als +/-5%

res1 pts: Enthält die Anzahl der zu vergebenden Punkte für die richtige Antwort.

Bonuspunkte-EXCEL-Tabelle:

Sollte es für die Prüfung Bonuspunkte geben, bspw. durch vorherige Leistungen im Laufe des Semesters, können diese über eine Bonuspunkte Tabelle berücksichtigt werden. Auch der Name dieser Datei kann bei den Konstanten zu Beginn definiert werden. Der Standardname lautet "Bonuspunkte.xlsx"

Die Bonuspunkte Tabelle hat dabei folgendes Datenformat:

```
-----
```

Benutzername	Name	Bonuspunkte
student	Student, Sven	2
student2	Studentin, Svenja	2

```
-----
```

PSSO-Tabelle

Weiterhin kann das Tool die aus dem PSSO exportierte Teilnehmerliste verwenden. Der Standardname der Datei lautet "PSSO_Teilnehmer.xls" und sie hat das folgende Dateiformat:

```
-----
```

: 3x Ignorierte Zeilen

:

mtknr	sortname	nachname	vorname	...
12345678	Student,Sven	Student	Sven	...
13456789	Studentin,Svenja	Studentin	Svenja	...

:

```
-----
```

„sortname“ wird nicht verwendet, da dieser bei langen Namen abgeschnitten wird und nicht mehr zwingend eindeutig ist. Stattdessen wird der verwendete Name aus den Spalten "nachname und "vorname" zusammengesetzt.

Der Zwischenschritt über die Eingabe der PSSO-Tabelle ist notwendig, da sich das einlesen der Matrikelnummer aus einer Freitextaufgabe als unzuverlässig erwiesen hat.

Export-Excel-Datei:

Die Ergebnisse des Tools werden im Anschluss als Excel-Datei exportiert. Der Name der Datei kann als Konstante am Anfang des Codes festgelegt werden, und lautet im Standardfall "Python_Testergebnisse.xlsx". Hierbei gilt zu beachten, dass:

- Alle Ausgaben werden in ein Blatt geschrieben
- Alle Daten eines Teilnehmers entsprechen einer Zeile
- Die Spalten werden nach folgendem Schema aufgeteilt:

Nr Name Vorname Familienname MatNr Note GesPkt BonusPkt A1Pkt ... A40Pkt ...

... A1_ID A01_Formel A01_Tol A1_ResRef A1_Res A1_v1 A1_v2...A1_v10 ...

...

... A40_ID A01_Formel A40_Tol A40_ResRef A40_Res A40_v1 A40_v2...A40_v10

1.5 Beschreibung des Auswertetools

1.5.1 Was ist Item Response Theory?

In der Item Response Theory (IRT) geht es um die Quantifizierung nur schwer messbarer latenter Merkmale (wie Intelligenz oder Fähigkeit) im Kontext der Klausurstellung und gehört zum Fachbereich der Psychometrie. Fragen die mithilfe der Methodik beantwortet werden sollen, reichen von „Wie hoch ist das Fertigungsmaß des Studenten?“ über „Wie schwer ist die Frage bzw. der gesamte Fragenpool?“ und „Ist der Fragenpool ausreichend, um die Studenten angemessen zu prüfen?“. Dabei wird ein probabilistischer Bezug von (richtiger) Antwort zu den gesuchten latenten Variablen hergestellt. Die Grundannahme lautet, dass Menschen mit hohem Fertigungs-Merkmal, auch eine höhere Chance haben richtig zu antworten. Den Zusammenhang von Fertigkeit zu einer richtigen Antwort zu messen, ist das Ziel der IRT.

Die folgende Abbildung zeigt eine Frage mit 3 Antwortmöglichkeiten. P ist dabei die Wahrscheinlichkeit das eine dieser Antworten ausgewählt wird, in Abhängigkeit zum Fertigungsmaß des Studenten (θ).

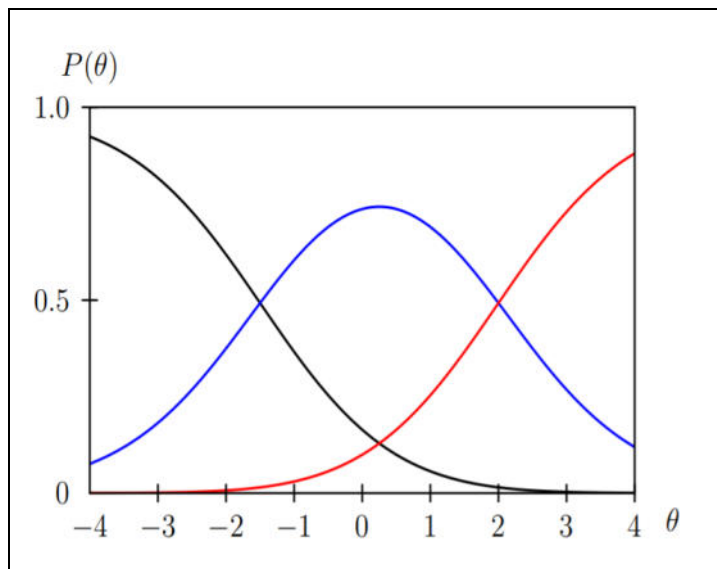


Abbildung 79: Wahrscheinlichkeitsfunktionen einer Antwort in Bezug zum Fertigungsmaß [1]

Bei der schwarzen Linie handelt es sich um die falsche Antwort. Die blaue Linie spiegelt ebenfalls eine falsche Antwort wider, jedoch wird fortgeschrittenes Wissen benötigt, um sich für diese Antwort zu entscheiden. Die rote Linie, gehört zu der Wahrscheinlichkeit einer richtigen Antwort und steigt, mit zunehmenden Fertigungsmaß θ des Studenten. Die Summe aller P 's ergibt je θ immer 1!

Im einfachsten Fall eines IRT-Modells, dem sogenannten Ein-Parameter Modell, ist die Wahrscheinlichkeit einer richtigen Antwort allein abhängig von Theta und der Verschiebung der Funktion entlang der X-Achse, was der Fragen-Schwierigkeit entspricht, wie folgende Darstellung zeigt.

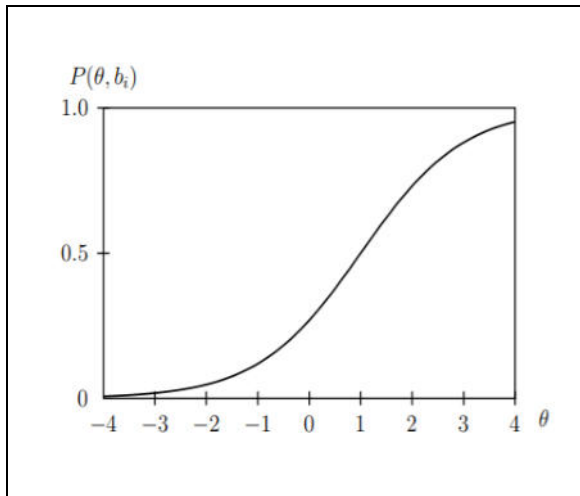


Abbildung 80: Ein-Parameter Modell [1]

Die Fragen-Schwierigkeit wird mit dem Parameter „b“ beschrieben. „b“ kann über mathematische Verfahren, bspw. Downhill-Simplex Methoden angenähert werden.

Insgesamt wurde das Ein-Parameter Modell um drei weitere Parameter ergänzt, um folgenden Fällen gerecht zu werden:

In dem Fall, das beispielsweise nur ein einfacher Fakt bekannt sein muss, um auf die richtige Lösung zu kommen, muss sich auch die Steigung des Graphen verändern. (Diskriminanzvariable „a“). Selbst ein Student ohne Hintergrundwissen, wird in einem Multiple Choice Test gelegentlich die richtige Antwort treffen. Die Wahrscheinlichkeit einer richtigen Antwort wird (im Multiple Choice Test) somit nie Null. (Pseudo-Rate Wahrscheinlichkeit „c“)

Selbst der beste Student mit exzellentem Wissen wird einen Fehler machen, bspw. aus Flüchtigkeit. Die Wahrscheinlichkeit einer richtigen Antwort wird nie zu Eins. (Unaufmerksamkeitsfaktor „d“) Der Graph des Vier-Parameter Modells ist in der folgenden Abbildung inkl. Der Beschreibungen dargestellt.

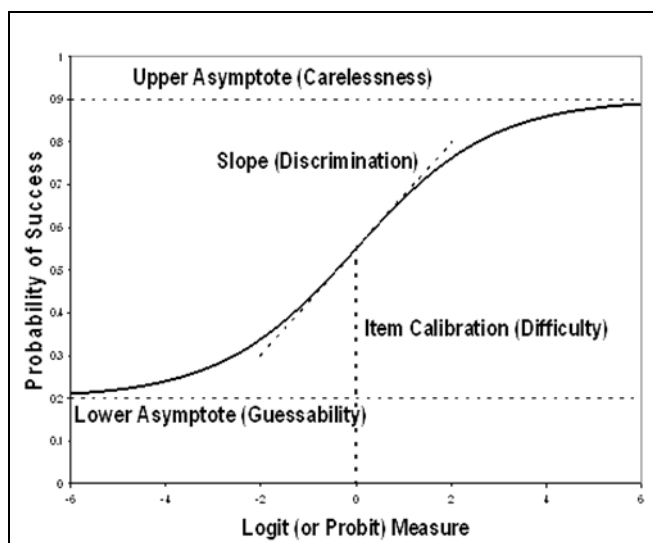


Abbildung 81: 4-Parameter Modell [1]

Aus den beschriebenen Zusammenhängen ergeben sich mithilfe der IRT-Methode verschiedene Nutzungsmöglichkeiten, wie bspw. die Bewertung der Schwierigkeit von Fragen oder der Einschätzung des Wissensstandes eines Studenten. Über viele Fragen hinweg, lässt sich auch die Gleichmäßigkeit der Fragen innerhalb eines Fragenpools quantifizieren. Unregelmäßigkeiten in den errechneten Variablen, können auch auf Fehler in den Aufgaben zurückzuführen sein, und ein weiterer Schritt in der Qualitätssicherung einer Klausur sein.

Weiterhin können mit genügend Testdaten auch die Fragenpools feinjustiert werden, um die erwartete Punktzahl an die Fähigkeit eines Studenten anzupassen. Insgesamt kann auch der Informationsgehalt einer Klausur ausbalanciert werden.

1.5.2 Nutzung des IRT-Tools

Das Tool ist in Form eines Jupyter-Notebooks verfügbar und benötigt eine Datenmatrix mit den Einzelergebnissen jedes Studenten pro Frage. Diese Datei wird im Nachkorrekturtool bereits erzeugt und in dem gleichen Ordner gespeichert. Vor dem Einlesen müssen allerdings noch die erste Zeile sowie die erste Spalte der Excel-Datei gelöscht werden, da diese nur die Indizes enthält. Mithilfe des Tools können die verschiedenen Parameter ausgegeben und eine Heatmap über die Antwortwahrscheinlichkeiten je Frage erzeugt werden. Weiterhin wurde ein Filter programmiert, welcher die einfachsten und schwierigsten Fragen erkennt, und Hinweise darauf liefern kann in welchen Fragen Probleme aufgetaucht sein könnten. Neben den IRT-Funktionen, können auch praktische Funktionen wie die Erstellung von Heatmaps für richtige und falsche Antworten erzeugt werden, wie in der folgenden Abbildung zu sehen.

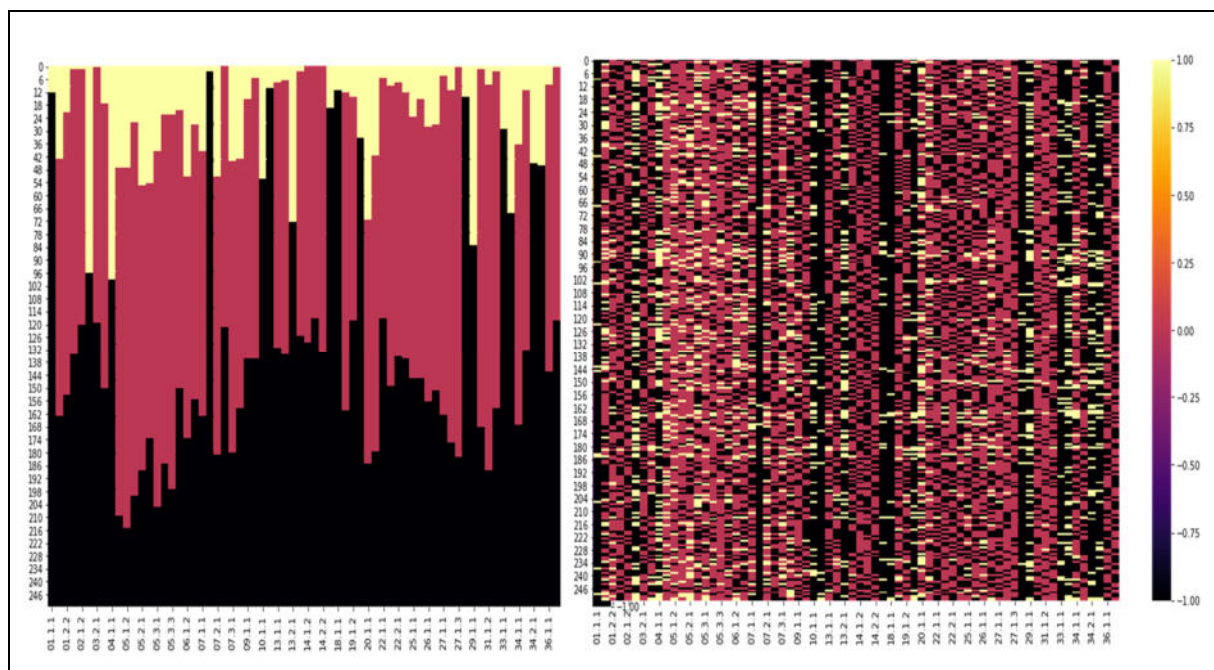


Abbildung 82: Heatmap richtiger (gelb), falscher (schwarz) und nicht gestellten (rot) Aufgaben je Student (rechts) und sortiert (links)

Weitere Funktionalitäten sind die Darstellung der 4 Parameter des IRT-Modells je Frage sowie die Thetas je Student. Außerdem ermöglicht das Tool die Generierung und Darstellung der IRT-Funktionen, sowie die zugehörigen Informationskurven je Frage.

Weiterhin wurde über den Machine-Learning Algorithmus „K-Means“ eine automatisierte Fehlerklassifizierung implementiert. Neben der graphischen Darstellung in einem Histogramm ist die tabellarische Form gut geeignet um häufige Fehler in einer Aufgabe zu entdecken. Als ersten Schritt wird dazu das richtige Ergebnis durch die Antwort des Studenten geteilt. Der K-Means Algorithmus gibt dann beispielweise folgende Tabelle aus:

Per K-Means gefundene Fehlercluster

Cluster ID	Abweichung	Anzahl
0	1,000	23
1	1.000,000	1
2	-1,000	15
3	5,760	1
4	-3,240	1
5	-0,001	1
6	0,100	1
7	0,001	3

Tabelle 33: Ausgabe des K-Mean Algorithmus

Durch die Division des richtigen Ergebnisses mit der Antwort des Studenten bedeutet eine 1 somit eine richtige Antwort. Dementsprechend haben 23 Studenten die Frage richtig beantwortet. Interessant ist nun, dass der häufigste Fehler ein Vorzeichenfehler war, den insgesamt 15 Studenten begangen haben. Häufige andere Fehler sind demnach Einheitenfehler, die an dem Faktor 1000 / 0,001 erkennbar sind.

1.5.3 Erkenntnisse aus dem Prozess

Als Auszug aus den Möglichkeiten der Nutzung des Tools dient das folgende Kapitel. Als ersten Eindruck, wie belastbar die Daten aus dem IRT Tool sind, zeigt Abbildung 83 den direkten Vergleich einer richtigen Antwort mit den Kenngrößen der IRT-Berechnung.

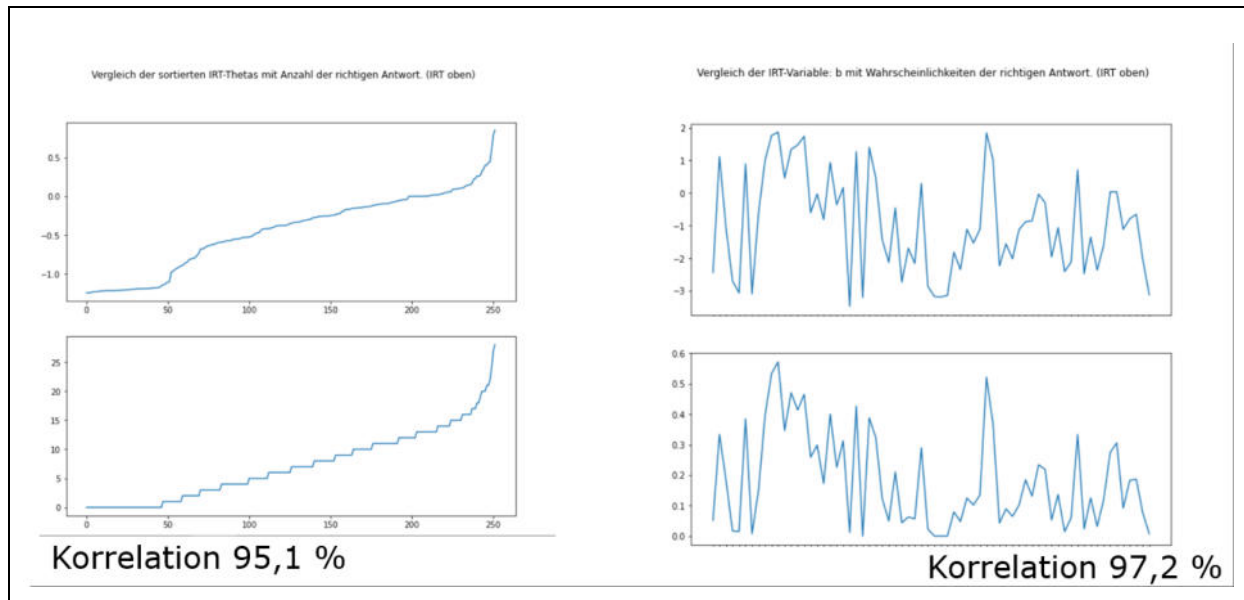


Abbildung 83: Intuitiver Vergleich der richtigen Antworten zu den Ergebnissen der IRT-Berechnung

Auf der linken Seite des Bildes sind oben die Thetas (also das Fertigungsmaß eines Studenten) und unten die Anzahl der richtigen Antworten eines Studenten geplottet. Man sieht, dass das Fertigungsmaß steigt, auch wenn sich in einer Spalte die Anzahl der richtigen Antworten nicht weiter erhöht. IRT ermöglicht somit also eine feinere Einstufung, auch bei quantitativ gleichen Ergebnissen. Die Pearson-Korrelation beider Graphen beträgt 95,1%

Auf der rechten Seite des Bildes wird die Fragenschwierigkeit „b“ (oben) mit der Wahrscheinlichkeit einer richtigen Antwort (unten) je Frage verglichen. Auch hier sieht man eine deutliche Korrelation (97,2%) beider Darstellungen. Intuitiv lässt sich also festhalten, dass die IRT Methode zu ähnlichen Ergebnissen kommt.

Die folgende Abbildung 84 stellt eine Monte-Carlo Simulation dar, mit deren Hilfe ermittelt werden soll, wie viele Punkte ein Student eines bestimmten Fertigungsmaßes Theta im Durchschnitt in diesem Fragenpool erreicht hätte.

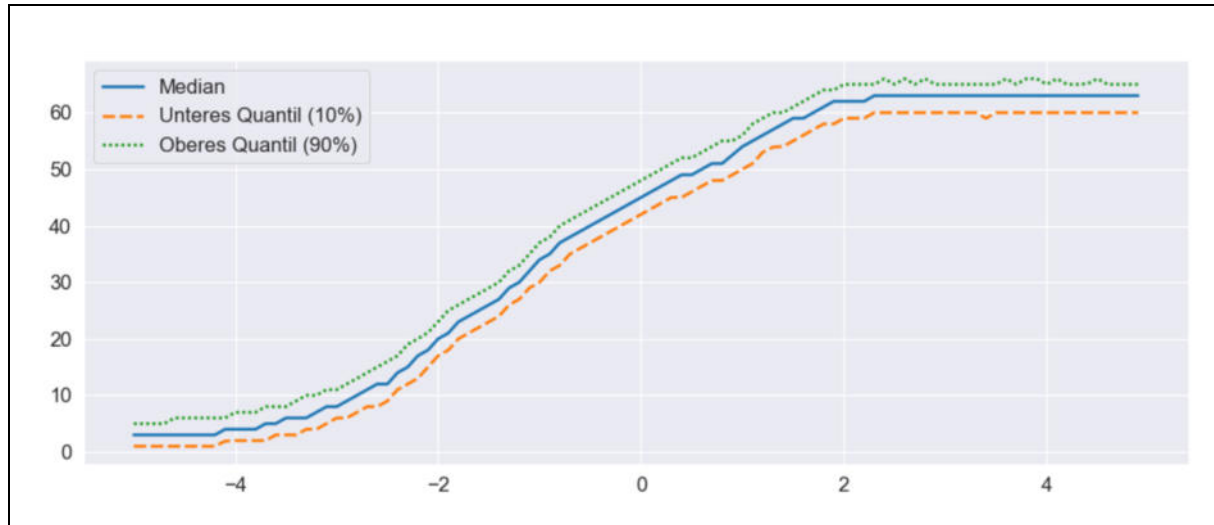


Abbildung 84: Monte Carlo Simulation je 1000 Studenten eines Fertigungsmaßes über den gesamten Fragenpool

Wie aus der Abbildung hervorgeht, würden Studenten mit dem Fertigungsmaß „0“ rund 45 Punkte erzielen. Da die IRT-Methodik auf Wahrscheinlichkeiten beruht, kann das 10% sowie das 90% Quantil mit rund 49 Punkten sowie respektive 41 Punkten ebenfalls angegeben werden. Ausreißer nach oben und unten sind möglich, aber in der Grafik nicht dargestellt.

Ein weiterer Vorteil in der Quantifizierung der Fragenpools ist, dass randomisierte Fragenpools miteinander verglichen werden und miteinander abgestimmt werden können. Wieso das notwendig ist, zeigt folgende Abbildung 85.

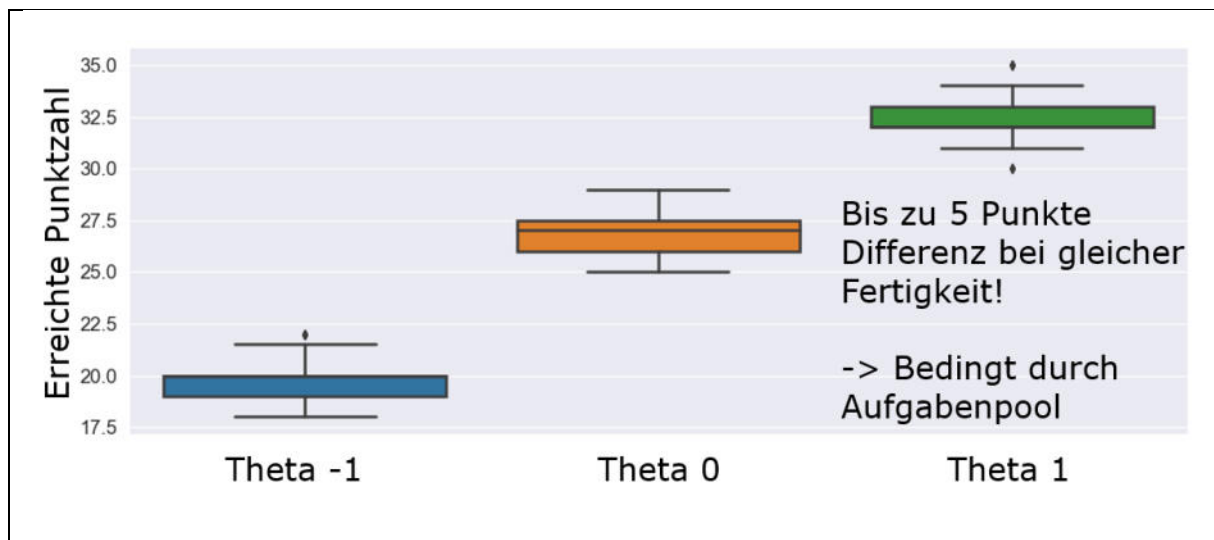


Abbildung 85: Abweichung verschiedener Fragenpools innerhalb des gleichen Tests

In der Abbildung ist zu sehen, dass obwohl es sich um den gleichen Test handelte, die zufällig ausgewählten Fragenpools zu Differenzen von bis zu 5 Punkten im Median führen können, obwohl die Studenten das gleiche Fertigungsmaß besitzen. Ermittelt wurde diese Abweichung ebenfalls über eine Monte Carlo Simulation, in der 1000 virtuelle Studenten einen Fragenpool bekamen und der Median jedes dieser Ergebnisse verglichen wurde.

2 Abbildungsverzeichnis

Abbildung 1: Darstellung des Programms auf Github	5
Abbildung 2: Programm von Github herunterladen	6
Abbildung 3: Inhalt des entpackten ILIAS-Aufgabengenerators	6
Abbildung 4: Anaconda Setup – Start (1/8).....	7
Abbildung 5: Anaconda Setup – Lizenzbedingungen (2/8)	7
Abbildung 6: Anaconda Setup – Installationstyp (3/8).....	7
Abbildung 7: Anaconda Setup – Speicherort (4/8).....	7
Abbildung 8: Anaconda Setup – Erweiterte Optionen (5/8)	8
Abbildung 9: Anaconda Setup – Abgeschlossen (6/8).....	8
Abbildung 10: Anaconda Setup – Info JetBrains (7/8)	8
Abbildung 11: Anaconda Setup – Ende(8/8)	8
Abbildung 12: Externe Bibliotheken.....	8
Abbildung 13: Öffnen der Konsole - Anaconda Prompt.....	9
Abbildung 14: Anaconda Prompt - Verzeichnis wechseln.....	9
Abbildung 15: Anaconda Prompt – Installation Paket 1 -> pip install tkScrolledFrame-1.0.4-py2.py3- none-any.whl.....	9
Abbildung 16: Anaconda Prompt – Installation Paket 2 -> pip install Pmw-2.0.1.tar.gz.....	9
Abbildung 17: Übersicht Anaconda Navigator	10
Abbildung 18: Spyder - Neues Projekt erstellen	10
Abbildung 19: Spyder - Projektname & -verzeichnis.....	10
Abbildung 20: Spyder – Ansicht -> leeres Projekt	11
Abbildung 21: Spyder - Ansicht -> Vollständiges Projekt	11
Abbildung 22: Spyder - ILIAS-Aufgabengenerator starten	11
Abbildung 23: GUI - Auswahl von Fragentypen	13
Abbildung 24: GUI (Formelfrage) - Hauptteil	13
Abbildung 25: GUI (Formelfrage) - Deklaration von zwei Variablen	14
Abbildung 26: GUI (Formelfrage) - Deklaration von drei Variablen.....	14
Abbildung 27: GUI (Formelfrage) - Ansicht von drei Variablen auf zwei geändert.....	14
Abbildung 28: GUI (Formelfrage) - Inhalt von Variable 3 wurde gelöscht	14
Abbildung 29: GUI (Datenbank) - Übersicht der Einträge (Beginn).....	15
Abbildung 30: GUI (Datenbank) - Übersicht der Einträge (Ende).....	15
Abbildung 31: GUI (Formelfrage) – Erstellung eines Fragentests	16
Abbildung 32: Ausgabe in Konsole nach erstelltem Fragentest.....	16
Abbildung 33:GUI (Formelfrage) - Erstellung eines Fragenpools.....	16
Abbildung 34: Ausgabe in Konsole nach erstelltem Fragenpool.....	16
Abbildung 35: GUI (Testeinstellungen) - Übersicht der Profile	17
Abbildung 36: GUI (Formelfrage) - Test erstellen mit Testeinstellungen	17
Abbildung 37: Excel-Vorlage (Formelfrage) mit definierten Fragen	18
Abbildung 38: Übersicht GUI für Formelfrage.....	19
Abbildung 39: GUI - Layout von Fragen-Text bearbeiten (Beispiel).....	20
Abbildung 40: Beispiel für Berechnung der Wertebereiche	20
Abbildung 41: Fragentext ohne Highlighting laden.....	21
Abbildung 42: Fragentext mit Highlighting laden.....	21
Abbildung 43: GUI (Formelfrage) - Wertebereich berechnen.....	23
Abbildung 44: Wertebereich für alle DB-Einträge berechnen (Warnung)	23
Abbildung 45: Wertebereich berechnen – Beispiel mit einer Formel.....	24
Abbildung 46: Wertebereich berechnen - Ausgabe auf der Konsole.....	24

Abbildung 47: Wertebereich berechnen - Beispiel mit vier Formeln.....	24
Abbildung 48: Wertebereich berechnen - Beispiel mit zuvor errechneten Ergebnissen.....	24
Abbildung 49: Auswahl des Fragenpools. Zielordner „anklicken“ und über „Ordner auswählen“ bestätigen.....	25
Abbildung 50: Inhalt des Zielordners	25
Abbildung 51: Darstellung Fragenpool-Taxonomien	26
Abbildung 52: GUI (Testeinstellungen) – 1/3	27
Abbildung 53: GUI (Testeinstellungen) – 2/3	27
Abbildung 54: GUI (Testeinstellungen) – 3/3	28
Abbildung 55: Fragen importieren	29
Abbildung 56: Import einer Excel-Datei - Fragen nicht in Datenbank vorhanden	29
Abbildung 57: Import einer Excel-Datei – Fragen in Datenbank bereits vorhanden (werden editiert).....	29
Abbildung 58: : Import einer Excel-Datei - Warnung bei doppelten Einträgen	29
Abbildung 59: Startseite Aufgabengenerator 2.0	64
Abbildung 60: Datenbank in Treeviewansicht.....	64
Abbildung 61: Variablen_Einfügen_UI für Formelfragen Variablen implementiert	65
Abbildung 62: Fragen GUI Layout für Formelfragen	66
Abbildung 63: Variablen-Interface für Singlechoice-Fragen	66
Abbildung 64: Variableninterface für Zuordnungsfrage	66
Abbildung 65: Erstellung table_index_list und table_index_dict.....	67
Abbildung 66: Erstellung index_list und index_dict.....	67
Abbildung 67: Auslesender Table-Reihennamen	67
Abbildung 68: Beispiel für Fragen erstellen und verwenden von String-Variablen	68
Abbildung 69: Beispiel für Übergabe in Dictionary	68
Abbildung 70: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 1/4	69
Abbildung 71: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 2/4	69
Abbildung 72: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 3/4.	69
Abbildung 73: Ausführbare Datei aus Programmcode erstellen - Virtual Environment aufsetzen 4/4	69
Abbildung 74: Ausführbare Datei aus Programmcode erstellen – Auflistung bereits installierter Bibliotheken	70
Abbildung 75: Ausführbare Datei aus Programmcode erstellen – Befehl zur Erstellung einer .exe.....	70
Abbildung 76: Ausführbare Datei aus Programmcode erstellen – Fehlermeldung wegen fehlender Bibliotheken	70
Abbildung 77: Ausführbare Datei aus Programmcode erstellen – Auflistung aller verwendeten Bibliotheken 1/2.....	71
Abbildung 78: Ausführbare Datei aus Programmcode erstellen – Auflistung aller verwendeten Bibliotheken 2/2.....	71
Abbildung 79: Wahrscheinlichkeitsfunktionen einer Antwort in Bezug zum Fertigungsmaß [1]	91
Abbildung 80: Ein-Parameter Modell [1]	92
Abbildung 81: 4-Parameter Modell [1]	92
Abbildung 82: Heatmap richtiger (gelb), falscher (schwarz) und nicht gestellten (rot) Aufgaben je Student (rechts) und sortiert (links)	93
Abbildung 83: Intuitiver Vergleich der richtigen Antworten zu den Ergebnissen der IRT-Berechnung	95
Abbildung 84: Monte Carlo Simulation je 1000 Studenten eines Fertigungsmaßes über den gesamten Fragenpool.....	96
Abbildung 85: Abweichung verschiedener Fragenpools innerhalb des gleichen Tests	96

3 Tabellenverzeichnis

Tabelle 1: Tabellenkalkulationsdatei - Bilder einbinden	31
Tabelle 2: Inhalt von ILIAS-Test und ILIAS-Pool	32
Tabelle 3: Darstellung von XML-Datei: Modules -> Test -> set_1 -> export.xml.....	33
Tabelle 4: Darstellung von XML-Datei: manifest.xml	33
Tabelle 5: Darstellung von XML-Datei: Modules -> Test -> set_1 -> export.xml (für einen Fragentest)	34
Tabelle 6: Darstellung von XML-Datei: Modules -> TestQuestionPool -> set_1 -> export.xml (für einen Fragenpool) – 1/2	35
Tabelle 7: Darstellung von XML-Datei: Modules -> TestQuestionPool -> set_1 -> export.xml (für einen Fragenpool) – 2/2	36
Tabelle 8: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 1/10.....	37
Tabelle 9: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 2/10.....	38
Tabelle 10: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 3/10.....	39
Tabelle 11: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 4/10.....	40
Tabelle 12: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 5/10.....	41
Tabelle 13: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 6/10.....	42
Tabelle 14: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 7/10.....	43
Tabelle 15: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 8/10	44
Tabelle 16: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 9/10.....	45
Tabelle 17: Darstellung von XML-Datei: qti.xml (für einen Fragentest) - Testeinstellungen – 10/10...	46
Tabelle 18: Darstellung von XML-Datei: qpl.xml (für einen Fragenpool).....	47
Tabelle 19: Darstellung der XML-Fragenstruktur für eine Formelfrage – Spezifischer Header – 1/4...	48
Tabelle 20: Darstellung der XML-Fragenstruktur für eine Formelfrage – Definition für Variablen – 2/4.	49
Tabelle 21: Darstellung der XML-Fragenstruktur für eine Formelfrage – Definition für Ergebnisse – 3/4	50
Tabelle 22: Darstellung der XML-Fragenstruktur für eine Formelfrage – Beschreibung Fragen-Text – 4/4	51
Tabelle 23: Darstellung der XML-Fragenstruktur für eine Singlechoice Frage – Spezifischer Header – 1/3	52
Tabelle 24: Darstellung der XML-Fragenstruktur für eine Singlechoice Frage – Definition für Antworten – 2/3	53
Tabelle 25: Darstellung der XML-Fragenstruktur für eine Singlechoice Frage – Definition Punktevergabe und Rückmeldung für Antworten – 3/3	54
Tabelle 26: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Spezifischer Header – 1/4	55
Tabelle 27: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Definition von Antworten – 2/4	56
Tabelle 28: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Definition Punktevergabe für Antworten – 3/4	57
Tabelle 29: Darstellung der XML-Fragenstruktur für eine Multiplechoice Frage – Definition Rückmeldung für Antworten – 4/4	58
Tabelle 30: Darstellung der XML-Fragenstruktur für eine Zuordnungsfrage – Spezifischer Header – 1/3	59
Tabelle 31: Darstellung der XML-Fragenstruktur für eine Zuordnungsfrage – Definition für Definitionen und Terme – 2/3	60

Tabelle 32: Darstellung der XML-Fragenstruktur für eine Zuordnungsfrage – Definition Punktevergabe und Rückmeldung für Definition/Term-Paare – 3/3	61
Tabelle 33: Ausgabe des K-Mean Algorithmus.....	94