

Technische Doku: LTI-Integration

Das Online-Übungssystem als LTI-Tool-Provider

Autor:

Immo Schulz-Gerlach, FeU-Softwaretechnik, ZMI

Version:

2.0.0 – 06. März 2025



Inhaltsverzeichnis

Einleitung	4
Zur Version 2.0.0 dieses Handbuchs / Änderungen ab Moodle 4.4	4
Ausgangssituation und Ziele	4
Weitere Vorbemerkungen	6
Implementierung eines LTI-1.1-Tool-Providers	7
Auswahl einer Java-Library zur Implementierung eines Tool-Providers	7
Eigene Erweiterung des Library-Forks	8
Weitere Entwicklungen auf Übungssystem-Seite	10
Juni 2019: Erweiterung um Aufgabenhefte als Tools	11
LTI-Launch und Autorisierung	14
LTI-Tool-Consumer beim Übungssystem anmelden	14
Username Parameter	16
Hinweis zu Moodle	16
GUID und »Protected«-Mode	17
Moodle-Toolkonfiguration	17
Startcontainer	19
Datenschutz-Einstellungen	19
Launch-URL in älteren Moodle-Versionen (vor Moodle 4.4)	20
Automatische Auswahl einer Tool-Konfiguration zu einem Launch-URL (vor Moodle 4.4)	20
Eventuell Future Work: Content Item Message	21
Autorisierung im Übungssystem-Kurs	21
Pflege der Whitelist	23
Implementierung der Whitelist	24
Autorisierung beim Launch-Request und in Folgerequests	25
Whitelist-Autorisierung	26
Student-Auto-Enroll oder lokale Autorisierung	26
Tutor-Auto-Enroll oder lokale Autorisierung	26
Session für Folge-Requests	27
Launch-Requests	27
Aufrufvarianten (Launch-URL und Custom-Parameter)	27
Referenz auf eine Aufgabe bzw. ein Aufgabenheft	28
Launch-URL bei Variante 1 (vor Moodle 4.4)	28
Launch-URL bei Variante 2 (ab Moodle 4.4)	28
Die Custom-Parameter bei Variante 2 (ab Moodle 4.4)	28
Ablauf eines Launch-Requests	30
Outcomes: Aufgabenergebnisse an Moodle übermitteln	33
Welche Ergebnisse können übertragen werden (und wann)?	33
Spezialfall Selbsttestaufgaben	35
Technische Realisierung der Outcomes-Serviceaufrufe	35
Erste Tripelklasse zur Assoziation von Gradebook-Entries zu Korrekturen	36
Zweite Tripelklasse zur Assoziation von Gradebook-Entries zu Aufgabenheft-Leistungen	36

LTI-Only-Mode	37
Optimistische Service-Calls und Sync-Jobs	39
Sync-and-Clean-Job	40
Embedded Views	43
(I)Frame-Modus	43
StudentenHeftStartSeite & StudentenHeftKorrekturSeite	43
Erweiterung des Embedding-Mechanismus des Online-Übungssystems	44
Einschränkungen	45
Query-Parameter	45
LTI-Return-URL	46
Return-Links derzeit nur im (I)Frame-Modus	46
Studentensicht	47
Einzelaufgabe	47
Aufgabenheft	49
Betreuersicht	51
Einzelaufgabe	52
Aufgabenheft	53
Wartungshinweise	55
DNS-Umbenennung eines Tool-Consumers	55
Änderung der Tool-Consumer-GUID	55
Änderung Outcomes-Service-URLs	56

Einleitung

Zur Version 2.0.0 dieses Handbuchs / Änderungen ab Moodle 4.4

Im März 2025 mit dem Update auf Version 4.4 wurde die LTI-Tooleinbindung in Moodle grundlegend geändert, was auch eine Überarbeitung der Übungssystem-Integration zur Folge hatte:

Zuvor erlaubte Moodle es, eine zentrale Toolkonfiguration im System zu hinterlegen, mit einem »Basis-URL« (Protokoll und Hostname des Tool Providers, hier OÜS), Key und Shared Secret sowie Einstellungen wie dem Erzwingen der Übertragung des Loginnamens. Und in den konkreten Kursumgebungen konnte man allgemein eine »Externes Tool«-Aktivität einfügen, dort einen mit dem OÜS-Hostnamen beginnenden, aber längeren Launch-URL einfügen, und Moodle erkannte dann, dass dieser URL zur hinterlegten Konfiguration passte, und lud die nach (wobei man einige Konfigurationen dann auch überstimmen konnte).

Ab Version 4.4 wurde das drastisch "vereinfacht": Nun wird in Moodle zwar weiterhin eine zentrale Toolkonfiguration angelegt wie bislang, aber die »Basis-URL«-Einstellung wurde zur "absoluten" »Tool-URL«-Einstellung, die nicht mehr pro eingebundenem Tool ergänzt werden kann. Und die Einbindung in die Kursumgebungen ist entsprechend auch grundlegend anders: Dort wird als Aktivität nicht mehr allgemein ein »Externes Tool« ausgewählt (und ein Tool-URL eingefügt), sondern explizit eine (als sichtbar konfigurierte) zentrale Toolkonfiguration als einzufügendes externes Tool ausgewählt. Haben wir dort in Moodle also eine solche Toolkonfiguration namens »Online-Übungssystem« angelegt, kann der User genau diese auswählen und "das Online-Übungssystem" als Aktivität in seine Kursumgebung aufnehmen. Den Launch-URL kann er dort nicht mehr anpassen, sondern ausschließlich optionale LTI-Custom-Parameter hinzufügen. (Zu jedem Custom-Parameter der Form key=value, die dort in einer Textarea als ein Parameter pro Zeile eingetragen werden können, wird Moodle dann einen LTI-konformen HTTP-Post-Parameter custom_key=value an den Toolprovider übermitteln.)

Damit das funktioniert, musste das Online-Übungssystem also so angepasst werden, dass es neben *kursspezifischen* Launch-URLs nun auch einen "globalen" Launch-URL versteht und in diesem Fall, also wenn die Aufgabenreferenz nicht im URL steckt, diese Infos aus den Custom-Parametern entnimmt.

Die Version 2.0.0 dieses Handbuchs wurde daher (neben diesem Einleitungs-Abschnitt) um eine Dokumentation dieser neuen Erweiterung ergänzt.

Ausgangssituation und Ziele

An der FernUniversität wird bereits seit Mitte der Neunzigerjahre das *Online-Übungssystem* bzw. dessen Vorgänger *WebAssign*¹ betrieben, mit dem insbesondere Einsende- und Selbsttestaufgaben online abgewickelt werden können. Später wurde zusätzlich die Moodle-Lernplattform an der FernUni in Betrieb genommen, mit der sich unter anderem auch Online-Aufgaben und -Tests anbieten lassen. Die Einsatzgebiete beider Systeme haben Überschneidungen, aber ohne dass dabei eines alle Features des anderen umfasst und dieses also obsolet macht.

So kann es insbesondere vorkommen, dass zu ein und demselben Kurs sowohl eine Moodle- als auch eine Übungssystem-Umgebung angeboten werden. In so einem Fall ist es verbreitet, dass die Moodle-Kursumgebung der "zentrale Anlaufpunkt" für die Studenten des Kurses ist, da sie mehr als nur die Übungsabwicklung bietet (z.B. Foren, News zur Kursabwicklung etc.), und in der Moodle-

07.03.25 Seite 4 / 57

Kursumgebung ein Link zur Ubungssystem-Kursumgebung gesetzt wird, ggf. auch umgekehrt. Allerdings navigiert ein Student über solche Links offensichtlich zwischen zwei Welten, mit separatem Login, unterschiedlichem Webdesign und ohne zentrale Zusammenfassung von Aufgabenergebnissen aus beiden Welten.

Das Ziel, dies zu verbessern, war einer der Ausgangspunkte der in diesem Dokument beschriebenen Entwicklung. Präziser wurden folgende Teilziele angestrebt:

- Statt von einer Moodle-Umgebung "nur" einen Link zu einer externen Übungssystem-Umgebung anzubieten, in welcher sich (alle oder weitere) Online-Aufgaben finden, sollten die im Übungssystem realisierten Aufgaben möglichst direkt in der Moodle-Kursumgebung zu öffnen und zu bearbeiten sein.
- Die sichtbaren Grenzen der Systeme sollten somit reduziert werden, die Aufgaben z.B. in einem Frame innerhalb der Moodle-Umgebung geladen werden.
- Insbesondere sollte dabei ein Single-Sign-On-Mechanismus zum Einsatz kommen, so dass ein bereits in Moodle eingeloggter Student zum Öffnen der Übungssystem-Aufgaben nicht erneut seinen Benutzernamen und sein Passwort eingeben muss.
- Für den Fall, dass auch in der Moodle-Kursumgebung selbst bestimmte Aufgaben angeboten werden, sollte es möglich sein, die Ergebnisse der Moodle- und der Übungssystem-Aufgaben zusammenzustellen, am besten im "Gradebook", das Moodle für jeden Kursteilnehmer führt.
- Da das ZMI nicht nur *eine* zentrale Moodle-Instanz betreibt, sondern für verschiedene Fakultäten getrennte Installationen, muss das Übungssystem mit mehreren verschiedenen Moodle-Instanzen zusammenarbeiten können und darf nicht auf genau einen Partner festgelegt werden.

Es gab im Wesentlichen zwei Realisierungsalternativen zu diesen Zielen:

- 1. Eine Eigenentwicklung auf der Basis u.A. von WebServices in Moodle und WebService-Clients im Übungssystem und dem ohnehin für die Zukunft geplanten Einsatz eines zentralen FernUni-Single-Sign-On-Systems (Shibboleth). Die Zusammenarbeit mit *mehreren* Moodle-Instanzen wäre dabei eine besondere Herausforderung: Soll z.B. das Übungssystem aktiv einen WebService von Moodle aufrufen, muss es dazu wissen, an *welche* Moodle-Instanz der Serviceaufruf zu richten ist.
- 2. Der LTI-Standard (Learning Tools Interoperability https://www.imsglobal.org/activity/learning-tools-interoperability) wird von Moodle bereits unterstützt und erfüllt obige Anforderungen. Es müsste also "nur" eine Funktion als LTI-Tool-Provider nachgerüstet werden. Der Umgang mit mehreren sog. Tool-Consumern (hier Moodle-Instanzen) und das Single-Sign-On über OAuth sind bereits im LTI-Standard geregelt. Für die Übermittlung von Aufgabenergebnissen aus dem Übungssystem an Moodle sieht LTI ab Version 1.1 einen "Outcome"-Service vor (für LTI 1.0 gab es schon eine Outcome-Extension), d.h. auch diese Möglichkeit kann über einen LTI-Tool-Provider (möglichst für LTI Version 1.1 oder neuer) realisiert werden, ohne selbst eine entsprechende WebService-Schnittstelle entwickeln zu müssen.

Unter der Voraussetzung, die LTI-Tool-Provider-Funktionalität nicht komplett selbst entwickeln zu müssen (nur mit der LTI-Spezifikation als Vorlage), sondern auf eine fertige Bibliothek zurückgreifen zu können, war für Alternative 2 ein geringerer Entwicklungsaufwand auf Seiten des Online-Übungssystems zu erwarten. Und auf Seiten von Moodle entfiele auf diesem Wege jegliche Entwicklung (im Gegensatz zur ersten Alternative). Auch ist LTI nicht Moodle-spezifisch, die zweite Alternative ist also auch flexibler für die Zukunft und ermöglicht die Einbettung von Übungssystem-Aufgaben auch in andere Systeme als Moodle. Daher wurde diese Alternative vorrangig untersucht – und erwies sich als gut realisierbar.

07.03.25 Seite 5 / 57

Der Einsatz von Übungssystem-Aufgaben als LTI-Tools ändert aber nichts am generellen Übungssystem-Workflow. Insbesondere sind die im Übungssystem eingestellten Bearbeitungstermine auch für den Einsatz der Aufgaben im LTI-Kontext bindend: Ruft ein Student von Moodle aus eine Übungssystem-Aufgabe per LTI auf, kann er den Aufgabentext erst ab dem im Übungssystem eingestellten *Bearbeitungsbeginn* überhaupt einsehen, und er kann seine Lösungen nur innerhalb der Bearbeitungsfrist (also bis zum *Bearbeitungsende*) einsenden. Danach kann er das Tool zwar weiterhin von Moodle aus aufrufen, aber seine Lösungen nur noch einsehen, dafür werden ihm – sobald sie bereit stehen – dort zusätzlich die Korrektur und ggf. die Musterlösung angezeigt (s. Studentensicht).

Weitere Vorbemerkungen

Wie bereits im vorhergehenden Abschnitt angesprochen, hält sich die Entwicklung der Tool-Provider-Schnittstelle im Online-Übungssystem – zumindest weitgehend² – an den LTI-Standard, sollte also mit verschiedenen Tool-Consumern zusammen arbeiten. Konkreter Einsatzkontext an der FernUni sind aber die verschiedenen ebenfalls vom ZMI betriebenen Moodle-Installationen. Wir haben daher auch in Tests nur Moodle als Tool-Consumer eingesetzt.

Angesichts dieser Ausgangslage soll im Folgenden oft kurz von "Moodle" an Stelle von "LTI-Tool-Consumer" die Rede sein, selbst bei allgemeinen Aussagen, die für alle möglichen Tool-Consumer gelten.

07.03.25 Seite 6 / 57

Implementierung eines LTI-1.1-Tool-Providers

Auswahl einer Java-Library zur Implementierung eines Tool-Providers

Um nicht alle technischen Aspekte zur Realisierung eines LTI-Tool-Providers aus der Spezifikation entnehmen und selbst implementieren zu müssen, wurde nach einer Library für Java (die Plattform des Online-Übungssystems) gesucht.

Ein wichtiger Aspekt, der insbesondere nicht selbst implementiert werden sollte, ist das Single-Sign-On über OAuth: Der User meldet sich nur beim Tool Consumer (i.F. Moodle genannt) an und soll bei Aufruf eines Tools (Übungssystem-Aufgabe) nicht erneut vom Übungssystem nach einem Login gefragt werden.

Die Auswahl an fertigen Java-Libraries zu diesem Thema ist recht überschaubar. IMS selbst verweist unter LTI – Sample Code http://www.imsglobal.org/learning-tools-interoperability-sample-code auf drei verschiedene Projekte:

- IMS Global LTI Utilities: Dies scheint wirklich eher eine kleine Utility-Sammlung als eine umfassende Lösung zu sein. Laut README des Github-Projekts wird speziell für "Tool Providers" i.W. folgende Funktionalität geboten: "Verifying an LTI launch request" und "Sending LTI 1.1 Outcomes request (xml-based)". Zu erstem Punkt scheint immerhin auch OAuth zu gehören. Im Quelltext findet sich auch ein Paket zu LTI2, aber auch das scheinen erstmal nur ein paar Toolklassen zu sein. Insgesamt schien mir dies allerdings etwas dünn zu sein, gerade auch im Hinblick auf die Dokumentation, wie man damit wirklich einen Tool Provider bauen kann.
- Tool Provider Class Library for Java: Unter diesem Link findet sich ein Java LTI Tool Provider package http://www.spvsoftwareproducts.com/java/lti_tool_provider/ (OSCELOT-Projekt http://projects.oscelot.org/gf/project/java_lti_t_prov/), das wirklich als komplette Library für die Implementierung eines LTI-Tool-Providers mit Beispiel-Projekt als Vorlage daherkommt. Neben der Behandlung von Launch-Requests und dem Aufruf von Outcomes-Services umfasst diese Library u.a. eine Verwaltung von Tool-Consumern und die Persistierung diverser, z.B. per Launch-Request empfangener Daten zu LTI-Contexten, Usern etc. in einer relationalen Datenbank. Die Library ist aber limitiert auf LTI Version 1.1 und wird anscheinend nicht mehr weiterentwickelt für LTI 2.
- TSUGI A Framework for Building Learning Tools (Java version): Dies wiederum scheint ein komplexeres Framework für die Entwicklung neuer Lern-Tools zu sein, zu dem unter anderem auch LTI-Support gehört. Ob es sich auch eignet, "nur" LTI in ein bestehendes Projekt einzubinden, das nicht auf diesem Framework aufbaut, habe ich nicht weiter verfolgt.

Den Beschreibungen und ersten Untersuchungen nach scheint von diesen drei Projekten nur das zweite (Tool Provider Class Library) die Anforderungen zu erfüllen³. Zumindest verspricht es genau das Gesuchte: Eine einfache Library zur Einbindung in ein Projekt, das LTI-Tool-Provider implementieren will, ohne gleich ein ganzes Framework zu bieten, in das sich mein Projekt ggf. einbetten müsste. Das beiliegende Demo-Projekt ermöglicht sofortige Versuche und hilft bei der Realisierung eines Tool-Providers trotz ansonsten dürftiger Dokumentation. Damit bietet diese Library den einfachsten Einstieg.

Die Wahl fiel daher auf diese zweite Library. Jedoch wurde sie im Online-Übungssystem nicht einfach als fertige Jar-Library eingebunden, sondern der Library-Quelltext wurde mit in den Quelltext des gesamten Übungssystem-Projekts aufgenommen, somit ein *Fork* der Original-Library im

07.03.25 Seite 7 / 57

Übungssystem-Projekt eröffnet, der eigene Anpassungen ermöglicht – die auch nötig waren.

Eigene Erweiterung des Library-Forks

Es stellte sich nach ersten Gehversuchen mit dem Beispielprojekt sowie Blicken in den Source Code der Library recht schnell heraus, dass einige Erweiterungen der Original-Library erforderlich oder wünschenswert waren:

- Zunächst erfüllt der Quellcode der Library nicht meine Mindest-Qualitätsanforderungen. Insbesondere sinnvolles Exception-Handling findet in der Original-Library nicht statt:
 - Library-eigene Funktionen werfen z.B. niemals Checked Exceptions, sondern antworten oft nur mit einem Boolean-Flag, ob sie ihre Arbeit fehlerfrei erledigen konnten oder nicht. Das wiederum lässt zwar wenigstens erkennen, *dass* ein Fehler aufgetreten ist, hilfreiche Informationen, *welcher* Fehler und *wo* genau werden so aber nicht gegeben.
 - Bei Aufrufen von Methoden der Standard-Klassenbibliothek werden alle Checked Exceptions dieser Methoden abgefangen (catched), i.d.R. einfach mit leeren Catch-Blöcken, d.h. Fehler werden ignoriert. Im besseren Fall wurden die Exceptions wenigstens insofern behandelt, dass die Library-Methode, in der eine Checked Exception gefangen wurde, ein Fehler-Flag setzte (s.o.). Sinnvolle Informationen aus der Exception (wie Art des Fehlers) werden damit aber unter den Teppich gekehrt.
 - Ein komplettes Redesign der Library habe ich mir gespart, aber zumindest leere Catch-Blöcke mit Log-Anweisungen für die aufgetretenen Exceptions gefüllt (zur Erleichterung des Debuggings). Teilweise wurde bei Fehlern, die im Produktionsbetrieb durchaus zu erwarten sind (z.B. HTTP-Kommunikationsfehler), die Library so erweitert, dass das Übungssystem in die Lage versetzt wird, sprechende Fehlermeldungen auszugeben (und nicht nur Meldungen der Art "Es ist ein LTI-Fehler auftreten").
 - Speziell für den Outcomes-Service musste das Fehlerhandling deutlich verbessert werden, um statt eines einfachen Flags (Erfolg ja/nein?) bestimmte Fehlersituationen anhand eigener Exception-Klassen unterscheiden zu können, siehe Sync-And-Clean-Job.
 - Bei der späteren Anpassung des Launches mit generischem, nicht mehr aufgabenspezifischem Launch-URL und vielmehr Verwendung von Custom-Parametern für eine Aufgaben(heft)-Referenz musste das Erkennen und Bereitstellen der Custom-Parameter (Parsen dieser Parameter aus der HTTP-Post-Submission und deren Bereitstellung über eine API) komplett neu programmiert werden, weil der bestehende Code nicht nur unpraktisch und schlecht konzipiert war, sondern auch fehlerhaft war und im Endeffekt gar nicht funktionierte (sprich: mit der Original-Library konnten die Custom-Parameter gar nicht ausgelesen werden).
- Die Library ist auf Speicherung diverser Daten in einem relationalen Datenbankschema ausgelegt. Viele z.B. per LTI-Launch-Request bezogene Daten werden in eine Objektstruktur (Instanzen von Entitätsklassen aus der Library) abgebildet, welche wiederum in einer Datenbank persistiert werden können. Im Kontext der Kurs-Autorisierung benötigt das Übungssystem jedoch noch weitere Informationen, die in den Launch-Request-Daten zwar vorhanden sind, von der Library jedoch nicht in der benötigten Form persistiert wurden.
 - Daher wurden das Datenbankschema und die Entitätsklassen des Frameworks entsprechend erweitert, um alle Übungssystem-Anforderungen erfüllen zu können.
 - Es wurde eine neue Tabelle lti_optcontext samt Model-Klasse Context eingefügt. Ein solcher Context beschreibt de facto eine Moodle-Kursumgebung, die Angabe einer Context ID ist im Launch Request laut LTI-Standard optional, wird vom Übungssystem aber zwingend vorausgesetzt.
 - Damit ist es im Übungssystem möglich, ein Paar zwischen einem Übungssystem-Kurs

07.03.25 Seite 8 / 57

und einem LTI-Context zu speichern (siehe Klassendiagramm in Abschnitt Kurs-Autorisierung), um alle Moodle-User, die im jeweiligen LTI-Context (Moodle-Kurs) zugriffsberechtigt sind, auch als zugriffsberechtigt für die Aufgaben des assoziierten Übungssystem-Kurses einzustufen (Single-Sign-On-Autorisierung).

- Für die Persistierung der LTI-Entitäsklassen in den schon von der Library vorgesehenen Datenbanktabellen wurde fürs Übungssystem ein eigener sog. "DataConnector" angelegt.
 - Es war zwar schon ein JDBC-DataConnector vorhanden, jedoch war dessen SQL i.W. auf die mySQL-Datenbank ausgelegt und nicht Oracle-kompatibel,
 - außerdem griff der direkt per eigener JDBC-Connection und ohne weitere Synchronisierung auf die DB-Tabellen zu.
 - Der neu erstellte Übungssystem-Connector nutzt dagegen synchronisierte DB-Zugriffe über zentrale WebAssign-Model-Methoden und somit über zentrale DB-Connections des Übungssystems. Der SQL-Code wurde Oracle-kompatibel gemacht.
- Da das Übungssystem kein reines LTI-Tool ist, sondern mit eigener Datenhaltung arbeitet, benötigt es von einem per LTI "ankommenden" Studenten auch den Loginnamen (der wiederum die Matrikelnummer enthält). Der LTI-Standard gibt das nicht wirklich her: In den Standard-Feldern des LTI-Launch-Requests wird zwar eine User-ID mitgesendet, die aber nicht dem Loginnamen entsprechen muss (bei Moodle typischerweise eine interne User-ID, anhand derer Moodle selbst z.B. bei Outcome-Requests den User wieder identifizieren kann, aus dem sich aber sein FernUni-Account oder seine Matrikelnummer nicht ableiten lassen).
 - Moodle (und auch andere Tool-Consumer) sendet diese Information (Loginname) jedoch durchaus im Launch-Request mit. Nur der Key dieses Post-Parameters ist nicht im LTI-Standard festgelegt, sondern ist eine Tool-Consumer-spezifische Erweiterung (Extension).
 Bei Moodle lautet dieser Key z.B. ext user username.
 - Um die Implementierung nicht spezifisch auf Moodle festzulegen, muss bei der Tool-Consumer-Registrierung im Übungssystem daher der Consumer-spezifische Key mit konfiguriert werden. Die Datenbanktabelle und Entitätsklasse für Tool-Consumer wurde um ein entsprechendes Attribut erweitert.
 - Der Lauch-PostService wiederum kann anhand dieser Information den Usernamen aus dem Launch-Request auslesen. (Auch hierzu wurde wieder eine Methode der Library entsprechend erweitert.)
- Es stellte sich bei Tests heraus, dass die Library nicht in jedem Fall das korrekte Encoding UTF-8, wie vom OAuth-Standard https://tools.ietf.org/html/rfc5849#section-3.6> vorgeschrieben) zur Dekodierung der Post-Request-Parameter verwendete, sondern mitunter auf ein falsches Encoding zur Dekodierung zurückgriff, was bei Auftreten von Sonderzeichen im Request (z.B. bei Studenten, deren Vor- oder Nachname einen Umlaut enthält) dazu führte, dass die Validierung der OAuth-Nachrichtensignatur fehlschlug und ein LTI-Launch für solche User nicht möglich war.
 - Dieses Problem habe ich *nicht* im Library-Fork gelöst, sondern im Übungssystem-eigenen LTI-Launch-PostService, indem dieser der Servlet-API mitteilt, dass diese (sobald sie von der ToolProvider-Library oder der OAuth-Library nach den Request-Parametern gefragt wird) den Request grundsätzlich nach UTF-8 dekodieren soll:

```
request.setCharacterEncoding("UTF-8");
```

• Prinzipiell hätte man diese Zeile auch in den Library-Fork einfügen können. Aber einerseits wollte ich den nicht unnötig abhändern, andererseits ist es äußerst wichtig, dass dieser Aufruf vor dem ersten Lesezugriff per Servlet-API auf den Request passiert, und das lässt sich Übungssystem-intern am übersichtlichsten garantieren.

07.03.25 Seite 9 / 57

Es bleibt anzumerken, dass diese Tool-Provider-Library sich, wie gesagt, auf das nicht mehr ganz aktuelle Protokoll LTI 1.1 beschränkt. Derzeit genügt das unseren Anforderungen, von LTI 2 eingeführte Protokollerweiterungen werden vom Übungssystem vorerst nicht benötigt.

Das Übungssystem nutzt folgende LTI-Features (mit Hilfe dieser Library):

- LTI-Launch-Requests: Es wurde ein WebAssignPostService (Basisklasse aller Services zur Behandlung von POST-Requests in der WebAssign-Architektur) implementiert, der Launch-Requests nach LTI 1.1 entgegennimmt und mit Hilfe der Library parst. Das Servlet aus dem Sample-Projekt zur Library diente nur als sehr grobe Vorlage, in diesem Service findet sich aber überwiegend Eigenentwicklung, insbesondere die Übungssystem-spezifische Autorisierung auf Kursebene. Siehe auch: Ablauf eines Launch-Requests.
- LTI-Outcome-Service: Optional bietet das Übungssystem an, Aufgabenergebnisse (Prozentpunkte zu einer Aufgabe) über den Outcomes-Service an Moodle zu übertragen, um sie dort ins sog. Gradebook einzutragen. Es gab schon zu LTI 1.0 eine entsprechende Erweiterung, mit LTI 1.1 wurde so ein Service (mit leichten insb. syntaktischen Modifikationen) mit in den Standard aufgenommen. Die Library unterstützt beide Varianten⁴.

Weitere Entwicklungen auf Übungssystem-Seite

Neben der eigentlichen Einbindung der LTI-Bibliothek und deren Nutzung zur Kommunikation mit Moodle waren für die Realisierung der eingangs aufgeführten Ziele noch ein paar LTI-unabhängige Aufgaben im Übungssystem zu erledigen, die hier der Vollständigkeit halber kurz aufgeführt werden sollen:

- Sessions: Das Online-Übungssystem arbeitet normalerweise zustandslos, also ohne Sessions. Die Authentifizierung erfolgt nach dem HTTP-AUTH-Mechanismus (aka Basic Authentication), bei dem sich lediglich der Browser die Credentials (Username und Passwort) des Benutzers merkt und bei jedem Folgerequest wieder mitsendet. Für das Single-Sign-On im Rahmen von LTI werden dagegen Sessions benötigt: Beim Launch-Request wird dem Übungssystem versichert, dass der Aufrufer (Moodle) den Benutzer bereits authentifiziert hat, und es wird eine User-ID mit übertragen, die sich das Übungssystem merken muss, um den User in Folgerequests wiedererkennen zu können, ohne eine neue Authentifizierung zu verlangen. Auch für den (I)Frame-Modus wird die Session benötigt.
- *URL-Rewriting*: Damit der Session-Mechanismus selbst dann funktioniert, falls der Benutzer seinem Browser das Speichern von Cookies untersagt hat, wurde ein URL-Rewriting-Mechanismus zum Übertragen der Session-ID integriert. (Das ist insofern nicht ganz so wichtig, als dass Moodle selbst derzeit m.W. nicht ohne Cookies funktioniert, erhöht aber die Flexibilität, z.B. falls ein User Cookies zwar generell verbietet, aber (nur) für Moodle eine Ausnahme eingerichtet hat, oder falls das Übungssystem einmal mit anderen Tool-Consumern verbunden werden soll, die auch ohne Cookies auskommen.)
- Autorisierung: Übungssystem-Aufgaben sind keine zustandslosen Tools, die jedem Moodle-Nutzer angeboten werden dürfen, sondern benötigen eine zusätzliche Autorisierungsprüfung.
- Im Rahmen der Koexistenz von Session-basierten LTI-Logins und zustandslosen lokalen HTTP-AUTH-Logins wurde auch Wert auf eine saubere Trennung der Verfahren gelegt, insbesondere darauf, dass die Session-basierte Autorisierung keine HTTP-AUTH-Mechanismen mehr nutzt (also bei fehlender Autorisierung nicht mit HTTP-Response-Code 401 den Browser Übermittlung von HTTP-AUTH-Credentials bittet).
- (I)Frame-Modus: Spezielle "abgespeckte" Ansicht des reinen Aufgaben-Contents optimiert zur Darstellung in Frames, um Toolseiten als IFrames in Consumer-Seiten einbetten zu können,

07.03.25 Seite 10 / 57

siehe Embedded Views.

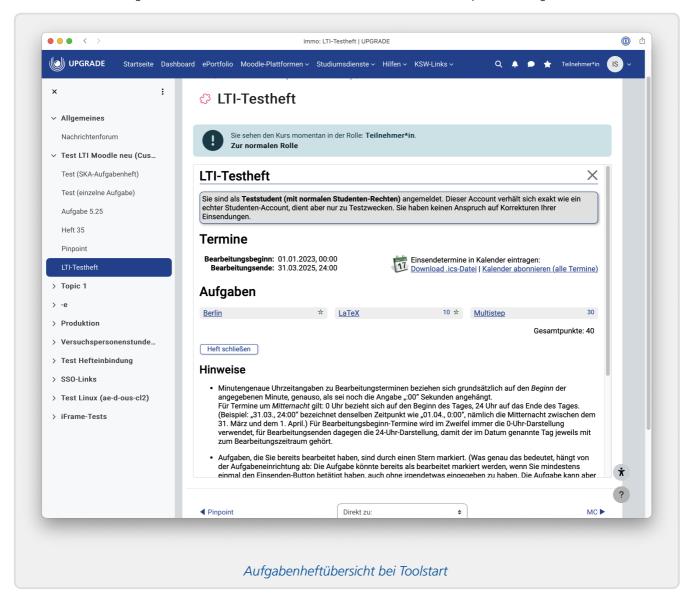
Juni 2019: Erweiterung um Aufgabenhefte als Tools

Im Online-Übungssystem sind alle Aufgaben in so genannten Aufgabenheften organsiert. Ein Aufgabenheft fasst also mehrere Aufgaben zusammen, ist aber mehr als nur ein "Ordner": Ein Aufgabenheft legt z.B. Bearbeitungstermine fest, einen Bearbeitungsbeginn-Termin, ab dem die Aufgaben des Hefts online eingesehen und bearbeitet werden können und einen Bearbeitungsende-Termin, bis zu dem die Aufgaben noch bearbeitet werden können, bzw. ab dem spätestens korrigiert werden kann und die Musterlösungen zu den Aufgaben – sofern vorhanden – online freigeschaltet werden. Weiterhin wird für jeden Teilnehmer aus den Einzelergebnissen zu den einzelnen Aufgaben auch ein Gesamtergebnis fürs Aufgabenheft berechnet.

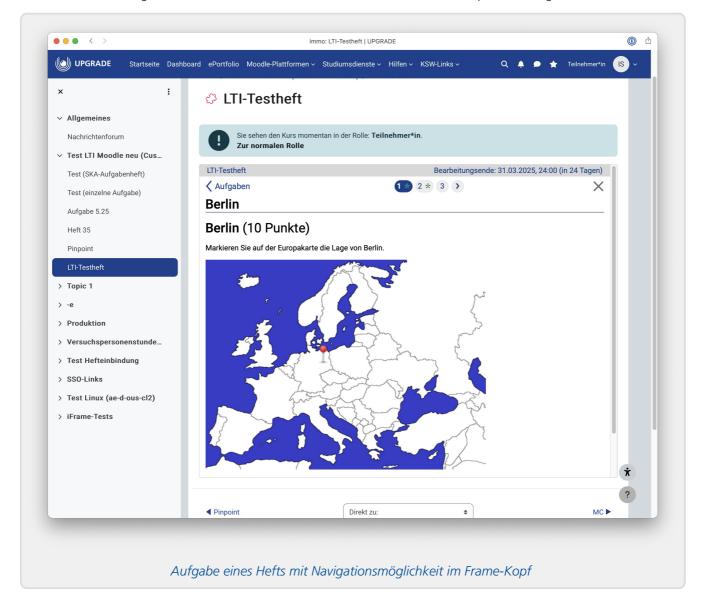
Bisher konnten aber nur einzelne Aufgaben als LTI-Tool in eine Lernplattform wie Moodle eingebunden werden. Bestand ein Aufgabenheft aus mehreren Aufgaben, die alle von den Studenten zu bearbeiten waren, musste jede Aufgabe einzeln als LTI-Tool eingebunden werden. Die Navigation innerhalb der Lernplattform musste dann über deren Menüs stattfinden, dafür gab's aber auch einen Gradebook-Eintrag pro Aufgabe (siehe Outcomes).

Ab sofort besteht alternativ die Möglichkeit, ein ganzes Aufgabenheft eines Online-Übungssystem-Kurses als LTI-Tool zu verwenden. Ein Teilnehmer bekommt dann beim Start des Tools (LTI-Launch) eine Heftübersicht mit den Bearbeitungsterminen angezeigt (die bei einzeln eingebetteten Aufgaben übrigens über ein Info-Icon im Kopf des Frames abrufbar sind) sowie eine Liste der Aufgaben mit Markierung bereits bearbeiteter Aufgaben zur Visualisierung des Bearbeitungsfortschritts. Innerhalb des Tool-Frames wird eine Navigation zu und zwischen den einzelnen Aufgaben des Hefts ermöglicht, wie die beiden nachfolgenden Screenshots beispielhaft zeigen:

07.03.25 Seite 11 / 57



07.03.25 Seite 12 / 57



07.03.25 Seite 13 / 57

LTI-Launch und Autorisierung

Das zentrale LTI-Feature ist der Aufruf eines Tools des Tool-Providers durch den Tool-Consumer, hier der Aufruf einer Übungssystem-Aufgabe (bzw. eines Aufgabenhefts) in einer Moodle-Kursumgebung. Die wesentlichen Mechanismen für diesen so genannten *Launch* eines Tools werden von der Tool-Provider-Library implementiert, auf Übungssystem-Seite musste noch ein Service zur Behandlung des eigentlichen POST-Requests implementiert werden, der u.a. das Parsen und Verarbeiten der im POST-Request gesendeten Daten an die Library delegiert.

Das LTI-Protokoll umfasst einen Single-Sign-On-Mechanismus auf OAuth-Basis, der es einem Studenten eines Moodle-Kurses ermöglicht, einfach Tools (hier: Aufgaben des Online-Übungssystems) zu benutzen, ohne sich erneut authentifizieren zu müssen. Auch das wird bereits von der TP-Library implementiert.

Da Übungssystem-Aufgaben jedoch keine allgemein zugänglichen Tools sind, die jeder Inhalteanbieter eines Tool-Consumers einfach all seinen Teilnehmern anbieten darf, war über diese von LTI garantierte *Authentifizierung* hinaus noch ein spezieller Mechanismus zur *Autorisierung* zu implementieren: Auf *welche* Übungssystem-Aufgaben darf ein Teilnehmer eines Moodle-Kurses denn überhaupt zugreifen? Dieser zentralen Frage widmet sich der dritte Abschnitt dieses Kapitels.

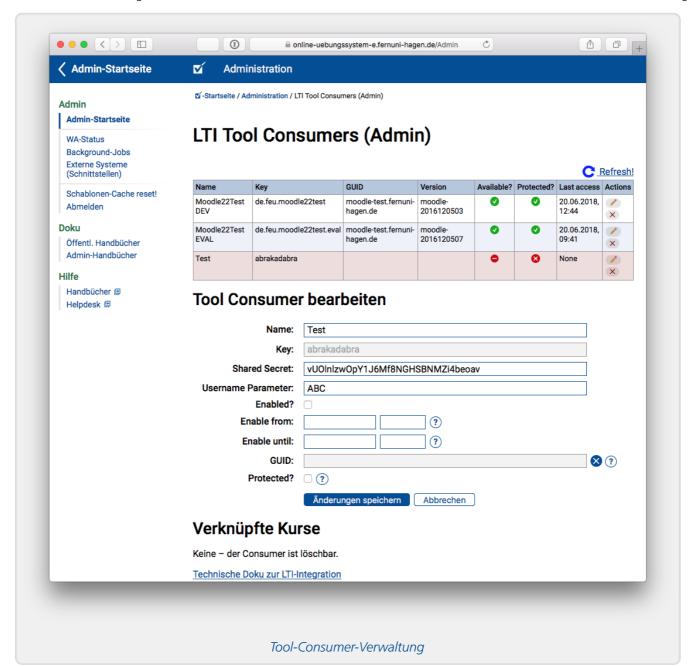
In den ersten beiden Abschnitten sei jedoch zuvor auf die – vom LTI-Standard bzw. OAuth geregelte – Verknüpfung von Tool-Consumer (Moodle) und Tool-Provider (Übungssystem) eingegangen. So kann das Übungssystem als Tool-Provider schließlich nicht jedem LTI-Launch-Request blind vertrauen, sondern der Aufrufer (Moodle) muss sich selbst beim Übungssystem zunächst "ausweisen". Die Systeme, die miteinander über LTI kommunizieren können sollen, sind vorab miteinander bekannt zu machen: Beim OAuth-Service-Provider / LTI-Tool-Provider (hier das Online-Übungssystem) ist vorab jeder Konsument (Service-/Tool-Consumer, hier Moodle) anzumelden. Darauf geht der erste Abschnitt dieses Kapitels ein, während der zweite dann die zugehörige Konfiguration der Gegenstelle (Moodle) betrachtet.

Abschließend folgt eine Zusammenfassung des im Online-Übungssystem implementierten Algorithmus' zur Abarbeitung eines LTI-Launch-Requests.

LTI-Tool-Consumer beim Übungssystem anmelden

Jede Moodle-Instanz, die autorisiert werden soll, Aufgaben des Online-Übungssystems einzubinden, ist vorab im Online-Übungssystem zentral zu erfassen (vom Admin). Die nachfolgende Abbildung zeigt einen Screenshot des UI, das weitgehend von der entsprechenden Oberfläche des Sample-Projekts zur Tool-Provider-Library entlehnt (aber im Übungssystem reimplementiert) wurde.

07.03.25 Seite 14 / 57



Die wichtigsten Elemente einer solchen Konfiguration (neben einem sprechenden Namen für das System) sind dort als »Key« und »Shared Secret« bezeichnet. Der »Key« (*OAuth Consumer Key*) muss ein eindeutiger, das Consumer-System identifizierender Schlüssel sein, der jedoch vom Übungssystem-Admin beliebig festgelegt werden kann, also nicht vom Consumer selbst vorgegeben wird. Das »Shared Secret« (*OAuth Consumer Secret*) ist eine Art Passwort, das dem Consumer-Admin zusammen mit dem Key mitzuteilen ist. (Es ist also ein "Geheimnis", das beiden Seiten, Consumer und Provider, bekannt ist.) Über dieses Shared Secret authentifiziert sich später das Moodle-System beim Übungssystem (wer das Secret nicht kennt, darf sich nicht mit dem Übungssystem unter diesem Key verbinden), es dient weiterhin zur Absicherung der Kommunikation, da der Consumer bei Launch Requests eine Signatur auf Basis des Shared Secrets erzeugt, die der Provider, der ebenfalls das Secret kennt, validieren kann. Details regelt das OAuth-Protokoll. (Da dieses Protokoll bereits von der verwendeten Library sowie Moodle implementiert wird, waren genauere Kenntnisse für die Implementierung der Tool-Provider-Schnittstelle im Übungssystem nicht nötig.)

07.03.25 Seite 15 / 57

Username Parameter

Eine Option wurde spezifisch fürs Online-Übungssystem hinzugefügt (siehe auch: Eigene Erweiterung des Library-Forks), und zwar die Einstellung »Username Parameter«: Das Online-Übungssystem benötigt in jedem Fall den LDAP-Namen eines Teilnehmers (derzeit der Form q plus Matrikelnummer) oder eines Betreuers. Ein solches Feld im Launch-Request ist im LTI-Standard nicht spezifiziert, konkrete Consumer-Implementierungen senden die Information aber in der Regel dennoch mit – nur eben in einem nicht standardisierten Parameter. Da sich der Name dieses Erweiterungs-Parameters von System zu System unterscheidet, ist er hier explizit anzugeben. Für Moodle lautet der korrekte Eintrag ext_user_username, bei anderen Systemen ist ggf. eine andere Einstellung nötig. Laut ExLibris-Group Knowledge Base

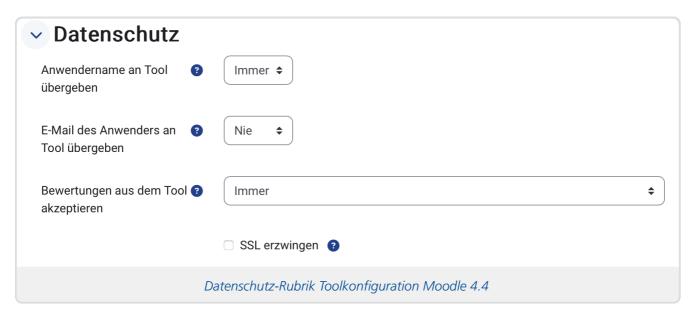
 lauten verschiedene Defaults:

- Moodle ext_user_username
- Sakai ext sakai eid
- Canvas lis user username
- Desire2Learn ext d21 username
- Other custom lis user username

Hinweis zu Moodle

Moodle sendet den besagen ext_user_username -Parameter nicht grundsätzlich mit, sondern nur, wenn in der Toolkonfiguration die Übermittlung des Usernamens explizit eingeschaltet wurde.

Bei Moodle 4.4 findet sich diese Einstellung z.B. in der Rubrik »Datenschutz« der LTI-Toolkonfiguration:



Die erste Einstellung in obigem Screenshot muss auf »Immer« lauten, damit der vom Online-Übungssystem zwingende benötigte Username-Parameter auch mitgesendet wird.

(Zu diesen Einstellungen siehe auch: Datenschutz-Einstellungen)

07.03.25 Seite 16 / 57

GUID und »Protected«-Mode

Laut LTI-Standard darf (optional, aber "recommended") ein Tool Consumer beim Launch-Request eine GUID mitsenden, mit der er sich identifiziert. Die Tool-Provider-Library speichert die GUID beim ersten Launch-Request automatisch. Sofern die Option »Protected« aktiviert wurde (siehe Screenshot oben), werden nachfolgenden Launch-Requests dann nur noch akzeptiert, wenn diese dieselbe GUID mitsenden.

In der Praxis mit Moodle hat sich allerdings ein Problem gezeigt: Moodle verwendet schlicht den Domain-Namen des Servers, genauer sogar den (falls Aliasse existieren), unter dem der Endnutzer im Browser die Moodle-Seiten aufgerufen hat. Der »Protected«-Mode funktioniert – zumindest bei Moodle – somit nur, wenn der Server nicht unter verschiedenen Domänennamen erreichbar ist – was aber bei HTTPS ohnehin (wegen des Domänennamens im Zertifikat) notwendig ist. Ändert man jedoch – wie bei den FernUni-Moodle-Installationen im Oktober 2017 geschehen – die Domain-Namen der Moodle-Installation, schlagen nachfolgende Launch-Requests zunächst fehl, da die GUID sich geändert hat.

Aus diesem Grund wurde dem UI eine neue Funktion zum Löschen der gespeicherten GUID hinzugefügt (vgl. (X) -Button in obigem Screenshot). Danach legt der erste nachfolgende Launch-Request die GUID neu fest.

Moodle-Toolkonfiguration

Wenn also zu einem Tool-Consumer (z.B. einer Moodle-Installation), der das Übungssystem als Tool-Provider einbinden darf, eine solche Konfiguration im Übungssystem erzeugt wurde, ist das vergebene Paar aus »Key« und »Shared Secret« Demjenigen mitzuteilen, der die entsprechende Konfiguration auf Consumerseite vornehmen soll.

In Moodle erfolgt dies durch Anlegen einer so genannten Tool-Konfiguration. Für das "Pairing" des Übungssystems mit einer Moodle-Installation sehen wir eine Moodle-globale Tool-Konfiguration vor. Diese kann nur von einem Moodle-Admin angelegt werden. Alternativ sieht Moodle auch für einen Betreuer die Möglichkeit vor – sofern vom Admin nicht deaktiviert –, eigene Tool-Konfigurationen anzulegen. Diese sind dann lokal für die Kursumgebung des Betreuers. Der folgende Screenshot zeigt beispielhaft das Anlegen einer lokalen Tool-Konfiguration in Moodle 4.4.

07.03.25 Seite 17 / 57

Lernumgebung Einstel	lungen	Teilnehmer/innen Bewertungen Berichte Mehr v	
Neues LTI ext	erne	es Tool hinzufügen	Alles aufklappen
Tool-Einstell	lung	en	
Name des Tools	?	Online-Übungssystem	
Tool URL	•	https://online-uebungssystem.fernuni-hagen.de/LTILaunch	
Tool-Beschreibung	•		
LTI Version	?	LTI 1.0/1.1 \$	
Anwenderschlüssel	3	some.moodle.key	
Öffentliches Kennwort	3	······································	
Angepasste Parameter	?		
			,
Standard-Startcontainer	?	□ Unterstützt Deep Linking (Content-Item Message) ③	
Inhalts-URL	•		
Mehr anzeigen			
> Services			
Datenschutz	<u> </u>		
Anwendername an Tool übergeben	?	Immer \$	
E-Mail des Anwenders an Tool übergeben	?	Nie 💠	
Bewertungen aus dem To akzeptieren	ool 🔞	Immer	
		□ SSL erzwingen	
		Tool-Konfiguration in Moodle	

Neben dem Key (in Moodle: »Anwenderschlüssel«) und Shared Secret (in Moodle: »Öffentliches Kennwort«) ist als zentrale Einstellung in Moodle der Launch-URL (in Moodle »Tool-URL« genannt) einzustellen (siehe Launch-Requests).

In früheren Moodle-Versionen nannte sich das URL-Feld noch »Basis-URL des Tools«. Da war es noch erlaubt, in den konkreten Tool-Einbindungen in den Lernumgebungen einen eigenen Tool-URL einzufügen, der nur mit dem Basis-URL hier begann, anhand dessen dann diese Toolkonfiguration

07.03.25 Seite 18 / 57

dem neuen Tool zugeordnet wurde. Ab sofort ist das nicht mehr möglich und dieser Tool-URL ist final, die Tooleinbindungen in den Lernumgebungen dürfen nur noch Custom-Parameter definieren, aber keinen abweichenden / um einen Path ergänzten Launch-URL. (Der Abschnitt Launch-URL in älteren Moodle-Versionen (vor Moodle 4.4) beschreibt i.F. der Vollständigkeit halber noch das Verfahren in älteren Moodle-Versionen, kann aber ignoriert werden, wenn ein neueres Moodle eingesetzt wird.)

Als LTI-Version ist 1.0/1.1 einzustellen, da die hier verwendete Library nur diese Versionen implementiert.

»Angepasste Parameter« werden hier in der zentralen Toolkonfiguration nicht benötigt, sondern später in der konkreten Einbindung in die Lernumgebungen ergänzt.

Startcontainer

Zu guter Letzt findet sich in der Toolkonfiguration noch eine Einstellung »Standard-Startcontainer« (s. Screenshot oben). Diese bestimmt die von Moodle einem Betreuer beim Einbinden des Tools vorzuschlagende »Startcontainer«-Einstellung, die dieser aber pro Tool-Ressource noch einzeln überstimmen kann. Der »Startcontainer« wiederum bestimmt, wie bzw. wo die vom Tool-Provider (Übungssystem) erzeugte Webseite vom Browser geöffnet werden soll: In einem neuen Browserfenster (als komplette Übungssystem-Webseite), im selben Browserfenster in Vollbild oder als eingebetteter Frame (Inline-Frame/iframe https://de.wikipedia.org/wiki/Inlineframe), siehe Kapitel Embedded Views.

Datenschutz-Einstellungen

Ein paar wichtige Einstellungen führt Moodle in der Rubrik »Datenschutz« auf:

- 1. Hier ist insbesondere die Option »Anwendernamen an Tool übergeben« einzuschalten. Damit wird aktiviert, dass Moodle den Loginnamen als separaten Username Parameter mit ans Online-Übungssystem überträgt. Dieser wird vom Online-Übungssystem zwingend benötigt.
- 2. Die zweite Einstellung regelt, ob auch die Mailadresse übertragen wird. Dies ist fürs Online-Übungssystem irrelevant: Wenn sie übertragen werden sollte, wird das OÜS sie nicht auswerten. Also kann die Option abgeschaltet bleiben.
- 3. Die Einstellung »Bewertungen aus dem Tool akzeptieren« legt fest, ob dieses Tool innerhalb von Moodle überhaupt Ergebnisse über das Outcomes-Feature entgegennehmen darf. Hier sollte am besten »immer« eingestellt werden.
 - Alternativ ist auch die "mittlere" Option »...an Dozierende delegieren« auswählbar, nach welcher dann pro Tool-Einbindung in eine Lernumgebung von den "Dozierenden" selbst eingestellt werden kann, ob Outcomes-Übermittlungen akzeptiert werden.
 - Aber da die Kontrolle über den Outcomes-Transfer ja schon im Online-Übungssystem getroffen werden kann, wäre eine zusätzliche Konfigurationsmöglichkeit pro Aufgabeneinbettung auf Moodle-Seite wohl eher eine unnötige Fehlerquelle: Falls jemand zwar im Online-Übungssystem die Outcomes-Übertragungen einschaltet, aber in Moodle dann das Akzeptieren von Bewertungen aus dem Tool *nicht* eingeschaltet hat, wird die Übermittlung nicht funktionieren außer eben, wenn hier zentral »immer« vorgegeben wird, dann können die Lehrenden das nicht mehr versehentlich abschalten (wohl aber immer noch die Outcomes-Übermittlung im Übungssystem abschalten).

Hinweis am Rande: Wenn zu einer Einbindung hier das Akzeptieren von Ergebnisübertragungen nicht eingeschaltet ist, legt das Online-Übungssystem beim Tool-Aufruf eines Studierenden auch *nicht* die ansonsten im Folgenden noch beschriebenen Datensätze zur

07.03.25 Seite 19 / 57

Verknüpfung seiner Übungssystem-Einsendungen mit seinem Moodle-Gradebook an.

Launch-URL in älteren Moodle-Versionen (vor Moodle 4.4)

Dieser Abschnitts beschreibt noch das Verfahren mit älteren Moodle-Versionen genauer und trifft auf Moodle ab Version 4.4 nicht mehr zu:

Wenn ein Betreuer in seine Moodle-Kursumgebung eine neue Aktivität »Externes Tool« (Element zur Einbindung eines LTI-Tools) hinzufügt, muss er zunächst einmal einen so genannten Launch-URL eintragen, in Moodle als »Start URL« [sic] bezeichnet. Dabei handelt es sich um den URL, an den der LTI-Launch-Request, ein HTTP-POST-Request zum initialen Toolaufruf, gesendet wird.

Das Übungssystem bietet den Kursbetreuern ein Interface, um gezielt eine Aufgabe ihres Übungssystem-Kurses auszuwählen, woraufhin es den entsprechenden Launch-URL zu dieser Aufgabe generiert, so dass der Betreuer ihn nur noch per Zwischenablage in die entsprechende Eingabezeile in Moodle kopieren muss.

(Zum Aufbau eines solchen Launch-URLs siehe: Launch-URL bei Variante 1 (vor Moodle 4.4).)

Automatische Auswahl einer Tool-Konfiguration zu einem Launch-URL (vor Moodle 4.4)

Moodle sucht nun bei Eingabe eines Launch-URLs automatisch nach einer zu diesem URL passenden Tool-Konfiguration. (Wenn es keine findet oder die gefundene nicht verwendet werden soll, kann der Betreuer selbst eine neue Tool-Konfiguration anlegen.)

Dazu enthält jede Tool-Konfiguration selbst den Beginn eines Launch-URLs als so genannten Basis-URL. Im einfachsten Fall ist dieser Basis-URL einfach nur der Hostname des Tool-Providers, dann gilt diese Tool-Konfiguration für alle Launch-URLs des Tool-Providers. Wenn dieser Hostname um weitere Pfadeinträge ergänzt wird, passt die Tool-Konfiguration nur auf entsprechende Launch-URLs, die mit den gleichen Pfadeinträgen beginnen.

Für die Kopplung des Übungssystems mit einem ZMI-Moodle sehen wir derzeit vor, dass in jeder Moodle-Installation zentral vom ZMI genau eine Tool-Konfiguration fürs gesamte Online-Übungssystem angelegt werden soll. In dieser wird als »Basis-URL des Tools« nur der Hostname online-uebungssystem.fernuni-hagen.de eingetragen. Moodle wird dann diese Tool-Konfiguration (mit globalen Standardeinstellungen) für alle Übungssystem-Launch-URLs vorschlagen.

Die Anbieter (Lehrende/Betreuer) müssen auf diese Weise nicht selbst Tool-Konfigurationen anlegen, sondern in allen Moodle-Kursumgebungen werden Übungssystem-Launch-URLs automatisch erkannt; die OAuth-Zugangsdaten (Key, Shared Secret) verlassen nicht das ZMI⁵.

Prinzipiell ist es natürlich auch möglich, speziellere Tool-Konfigurationen mit längeren Basis-URLs zu erstellen. Möchten wir z.B. für einen bestimmten Veranstalter xyz eine spezielle Tool-Konfiguration (z.B. mit anderen Standardeinstellungen) vorgeben, so können wir zusätzlich zur obigen allgemeinen eine neue mit Basis-URL online-uebungssystem.fernuni-hagen.de/xyz erstellen (vgl. oben beschriebenen URL-Aufbau). Falls eine speziellere und eine allgemeinere Tool-Konfiguration parallel in Moodle existieren, wird Moodle stets die speziellere vorschlagen (laut Moodle-Hilfetext zum Basis-URL-Eingabefeld).

07.03.25 Seite 20 / 57

Eventuell Future Work: Content Item Message

Ab Version 3 unterstützt Moodle die optionale LTI-Extension "Content Item Message" (i.F. kurz CIM genannt). Damit wäre es möglich, in Moodle eine zentrale Tool-Konfiguration anzulegen, die nicht mehr das manuelle Kopieren des Launch-URLs wie oben beschrieben durch den Betreuer erfordert.

Vielmehr ließe es sich so einrichten, dass der Betreuer in Moodle direkt eine Übungssystem-Toolkonfiguration auswählt und dann einen »Inhalt auswählen«-Button anklickt, woraufhin er direkt eine Übungssystem-Aufgabe zur Einbindung in Moodle auswählen könnte.

Das erfordert jedoch die Implementierung eines entsprechenden CIM-Services im Online-Übungssystem, der eine solche LTI-CIM-Anfrage versteht und eine vom Online-Übungssystem selbst erzeugte, in Moodle in einem I-Frame anzuzeigende UI zur Auswahl einer Aufgabe erzeugt (mit Autorisierung, d.h. der Betreuer darf nur "seine eigenen" Aufgaben auswählen können).

Die vorhandene Tool-Provider-Library unterstützt keine solchen CIMs, das Parsen einer entsprechenden LTI-Nachricht mit allen Aspekten wie insb. OAuth müsste also komplett selbst implementiert und im eigenen Library-Fork nachgerüstet werden. Die Behandlung von LTI-Launch-Services könnte ggf. als Vorlage dafür herangezogen werden, aber die Implementierung erfordert natürlich ein genaues Studium der CIM-Spezifikation.

Das Feature würde einen gewissen Komfortgewinn für die Betreuer bedeuten, andererseits wird derzeit ein relativ hoher Entwicklungsaufwand erwartet. Für Studenten bietet dieses Feature dagegen keinen Mehrwert, und auch die Einbindung von Aufgaben in Moodle erfolgt nicht so häufig, dass dieser Komfortgewinn erheblich ins Gewicht fiele. Daher ist noch offen, ob dies jemals realisiert wird.

Autorisierung im Übungssystem-Kurs

LTI mit seinem Single-Sign-On bietet eine zentrale *Authentifizierung*. D.h. Moodle erfragt die Credentials eines Benutzers und prüft, ob ein entsprechender Benutzeraccount für Moodle existiert und ob sich der Benutzer mit dem korrekten Passwort ausgewiesen hat.

Weiter prüft Moodle die Autorisierung für bestimmte Kursumgebungen, d.h. ein in Moodle eingeloggter Benutzer darf nur auf bestimmte Kursumgebungen innerhalb Moodles zugreifen. Enthält eine solche Kursumgebung eine Übungssystem-Aufgabe als "LTI-Tool", so leitet Moodle die Anfrage zum Starten/Bearbeiten der Aufgabe an das Übungssystem weiter und garantiert dabei dem Übungssystem praktisch, dass der eingeloggte Benutzer (dessen Accountname dem Übungssystem auch mitgeteilt wird) sich authentifiziert hat und autorisiert ist zum Zugriff auf die Moodle-Umgebung, in der die Übungssystem-Aufgabe eingebettet ist.

Das allein stellt aber nicht sicher, dass der Benutzer auch zum Zugriff auf die Übungssystem-Aufgabe selbst autorisiert ist. Nehmen wir beispielsweise an, ein Kursbetreuer eines Kurses A bindet in seiner zu Kurs A gehörenden Moodle-Umgebung den Launch-URL zu einer Übungssystem-Aufgabe eines Kurses B ein, der von einem ganz anderen Lehrgebiet veranstaltet wird. Dann sind in der Regel zwar alle Beleger von Kurs A zugriffsberechtigt für die Moodle-Kursumgebung zu A, aber i.A. nicht zugriffsberechtigt für die Aufgaben zu Kurs B, wozu eine Belegung von Kurs B nötig wäre.

Normalerweise ist im Online-Übungssystem die Autorisierung für Online-Übungen an die Kursbelegung (laut SLO-Datenbank) des Kurses (identifiziert anhand der Kursnummer) im entsprechenden Semester gebunden. Natürlich könnte man das auch bei der LTI-Integration einfach so belassen: Moodle würde nur garantieren, dass der Nutzer mit einer gegebenen Matrikelnummer sich korrekt eingeloggt hat (also authentifiziert ist) und nun auf die per LTI angebotene Aufgabe zugreifen möchte, und das Übungssystem könnte nun prüfen, ob der Nutzer überhaupt eine

07.03.25 Seite 21 / 57

Kursbelegung hat und andernfalls den Zugriff ablehnen.

Diese simple Lösung wäre aber ziemlich unflexibel. Es sind durchaus auch Konstellationen denkbar, in denen in einer Moodle-Kursumgebung die zugriffsberechtigten Teilnehmer nicht anhand der Belegung eines bestimmten Kurses (dem dann auch die einzubindenden Übungssystem-Aufgaben untergeordnet werden müssten) bestimmt werden. Z.B. könnten die Moodle-Kursteilnehmer manuell angemeldet werden oder Beleger unterschiedlicher Kurse sein.

Die LTI-Integration im Online-Übungssystem soll es *ermöglichen*, dass alle für die einbindende Moodle-Kursumgebung autorisierten Teilnehmer auch automatisch autorisiert für den Übungssystem-Kurs sein sollen. Da aber nicht – wie in obigem Beispiel skizziert – die Teilnehmer *jeder beliebigen* Moodle-Kursumgebung grundsätzlich für *jede beliebige* Übungssystem-Aufgabe selbst anderer Kurse und Lehrgebiete autorisiert sind, wurde im Online-Übungssystem für LTI ein neuer Autorisierungsweg implementiert: Zu einem Übungssystem-Kurs können ein oder mehrere Moodle-Kursumgebungen (allgemeiner: LTI-Kontexte) als autorisiert freigeschaltet werden. So kann in einem Online-Übungssystem-Kurs vermerkt werden, dass alle Teilnehmer einer bestimmten Moodle-Kursumgebung bei LTI-Zugriffen als autorisiert gelten, die Aufgabe also per LTI von dieser Moodle-Umgebung aus aufrufen und dazu einsenden dürfen. Im einfachsten Fall beziehen sich beide (Übungssystem- und Moodle-Kursumgebung) auf denselben FernUni-Kurs, aber das ist nicht notwendig.

Die Liste dieser zu einem Übungssystem-Kurs zugriffsberechtigten LTI-Kontexte wird projektintern – und auch in diesem Dokument – als *LTI-Whitelist* bezeichnet, im Betreuer-UI ist diese Whitelist mit »LTI-Verknüpfungen« überschrieben.

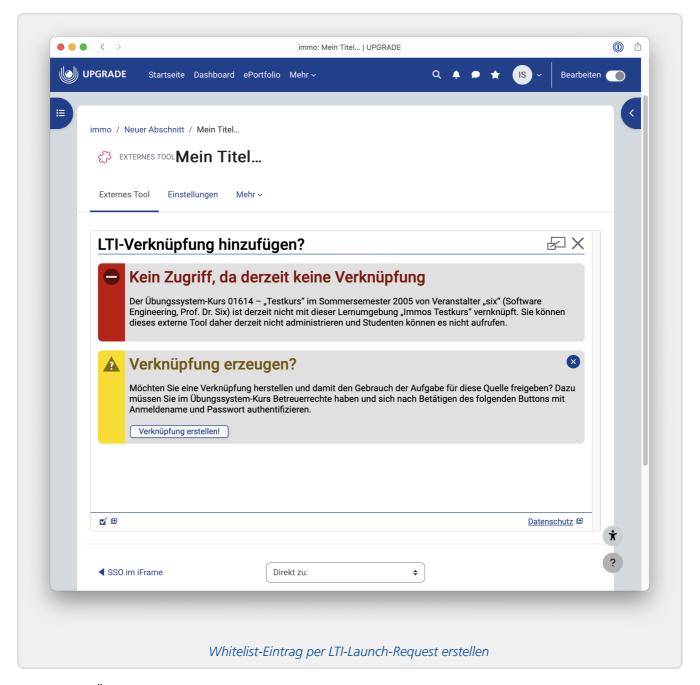
Per Default erlaubt jeder Whitelist-Eintag jedem vom Tool Consumer (Moodle) für den Kontext (Kursumgebung) autorisierten Teilnehmer den Zugriff auf alle im Kontext eingebetteten Aufgaben von Übungssystem-Kursen, in denen der Kontext auf der Whitelist steht. In Ausnahmefällen jedoch kann es durchaus gewünscht sein, dass das Übungssystem (in zweiter Instanz, also nach der Whitelist-Autorisierung) dennoch auch noch seine Standard-Autorisierung (z.B. Prüfung einer Kursbelegung) vornimmt. Daher gibt es zu jedem Whitelist-Eintrag eine (standardmäßig aktivierte) Option »Studentenrechte vererben«, durch deren Abschaltung sich die lokale Autorisierung optional zuschalten lässt.

Ein Eintrag auf dieser Whitelist erlaubt übrigens nicht nur den Teilnehmern (Studentenrolle) eines Moodle-Kurses den Zugriff auf die verknüpften Übungssystem-Aufgaben, sondern auch jeder Nutzer, der für die Moodle-Kursumgebung über Betreuerrechte verfügt, gilt im Übungssystem standardmäßig als autorisierter Betreuer, kann also z.B. von Moodle aus auch Änderungen an der Aufgabenstellung im Übungssystem vornehmen. Auch dies ist abschaltbar, über die Option »Betreuerrechte vererben« des jeweiligen Whitelist-Eintrags. Ist diese deaktiviert, werden LTI-Launches in der Betreuerrolle nur noch akzeptiert, falls zum selben Benutzernamen auch im Übungssystem-Kurs ein Betreuer angemeldet ist. Unter der Annahme, dass LDAP-Accounts (und keine lokalen Accounts) verwendet werden, bekommen dann also nur Mitarbeiter, die in beiden Rechten explizit die Betreuerrolle zugewiesen bekommen haben, per LTI-Launch in der Betreuerrolle auch vollen Betreuerzugriff im Übungssystem (mit Single-Sign-On). Ein Moodle-Betreuer, der im Übungssystem nicht als Betreuer angemeldet ist, erhält beim Launch-Request in Betreuerrolle eine Fehlermeldung, kann das Tool aber immer noch in der Studentenrolle aufrufen.

07.03.25 Seite 22 / 57

Pflege der Whitelist

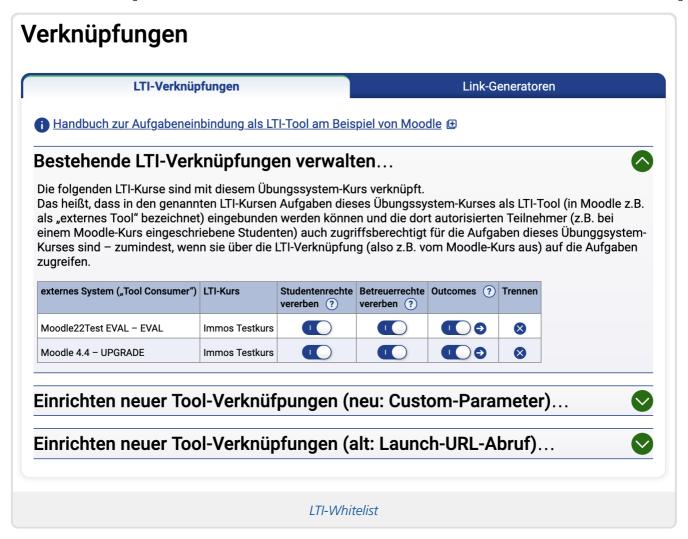
Die Erfassung neuer Verknüpfungen auf der Whitelist erfolgt über einen LTI-Launch-Request: Ein Betreuer loggt sich in seine Moodle-Umgebung ein und fügt dieser ein neues »externes Tool« hinzu. Dort trägt er den Launch-URL einer Übungssystem-Aufgabe eines Kurses ein, für den er im Übungssystem über (lokale) Betreuerrechte verfügt. Beim ersten LTI-Aufruf dieser Übungssystem-Aufgabe in seiner Betreuerrolle prüft das Übungssystem, ob die aufrufende Moodle-Umgebung bereits auf der Whitelist des Übungssystem-Kurses, zu dem die Aufgabe gehört, steht. Wenn nicht, wird der dem Betreuer angeboten, die Verknüpfung anzulegen, wozu er sich durch einen lokalen Betreuer-Login beim Übungssystem separat authentifizieren muss:



Nur wer im Übungssystem-Kurs lokal Betreuerrechte hat, kann also solche Verknüpfungen anlegen. Durch das Anlegen per LTI-Launch-Request ist keine komplexere Konfiguration wie manuelle Eingabe eines URLs o.ä. erforderlich.

Im Betreuermenü des Übungssystem-Kurses kann unter »LTI-Verknüpfungen« jederzeit die aktuelle Whitelist angezeigt werden, und dort können auch Whitelist-Einträge wieder gelöscht werden:

07.03.25 Seite 23 / 57

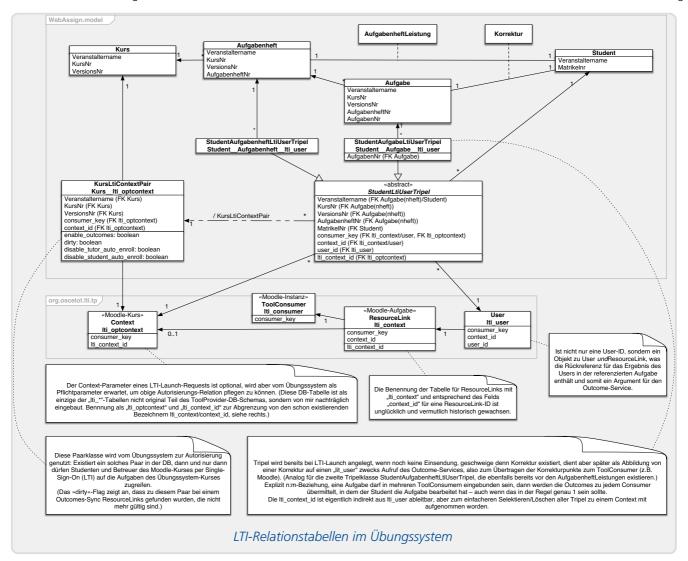


Wie auf obigem Screenshot zu sehen, findet sich im Whitelist-UI auch die Möglichkeit, das Outcomes-Feature (s.u.) zu aktivieren. Die Optionen »Studentenrechte vererben« und »Betreuerrechte vererben« wurde bereits im vorangehenden Abschnitt beschrieben.

Implementierung der Whitelist

Das folgende Klassendiagramm zeigt einen Ausschnitt des Übungssytem-Datenbankschemas. Jede Klasse steht für eine Model-Klasse mit zugehöriger Datenbanktabelle (für die Persistierung). Falls die Namen von Klasse und DB-Tabelle abweichen, werden beide genannt (zuerst der Klassenname, dann der Tabellenname). Die aufgeführten Attribute bilden den Primärschlüssel und im zweiten, normalerweise für Methoden vorgesehenen "Compartment" werden ggf. interessante Nicht-Schlüsselfelder aufgeführt.

07.03.25 Seite 24 / 57



Die obere Reihe (Kurs, Aufgabenheft, ..., Student) stellt Kerntabellen von WebAssign dar, die untere Reihe (lti_*) zeigt Tabellen zur LTI-Toolprovider-Library. (Die ganz linke davon, lti_optcontext, ist dabei nicht Teil der Original-Library, sondern wurde wegen Eigenbedarfs dem Übungssystem-eigenen Fork der Library hinzugefügt.) Vertikal dazwischen finden sich Relationstabellen, die Beziehungen zwischen diesen beiden Ebenen herstellen.

Die LTI-Whitelist wird durch die links abgebildete Relationsklasse KursLtiContextPair implementiert. Die zwei Nicht-Schlüsselfelder dieser Tabelle werden, ebenso wie die rechts davon abgebildeten weiteren Relationen, später im Kapitel zum Outcomes-Service erläutert.

Autorisierung beim Launch-Request und in Folgerequests

Ein Benutzer, z.B. ein Moodle-Kursteilnehmer, startet zunächst eine Übungssystem-Aufgabe (oder ein Aufgabenheft) über einen so genannten Launch-Request. (Er ruft das Tool in Moodle auf, Moodle öffnet eine Seite z.B. mit einem IFrame und sendet dann den Launch-Request an den Tool Provider, hier das Online-Übungssystem.)

07.03.25 Seite 25 / 57

Whitelist-Autorisierung

Beim Launch-Request erfolgt die "Erst-Autorisierung" des Users anhand der oben beschriebenen Whitelist. Ruft ein Teilnehmer die Aufgabe / das Heft auf, so wird überprüft, ob der Kurs dieser Aufgabe / dieses Hefts überhaupt für die Verwendung in dem aufrufenden Moodle-Kontext per Whitelist-Eintrag freigeschaltet wurde, und falls ja, gilt der für den Moodle-Kontext autorisierte Teilnehmer automatisch auch als im Übungssystem autorisiert. Ggf. werden dazu im Übungssystem noch lokale Account- und Belegungseinträge erzeugt (Auto-Enroll), siehe folgenden Abschnitt.

Student-Auto-Enroll oder lokale Autorisierung

Im Normalfall verlässt sich das Online-Übungssystem darauf, dass wirklich jeder Student, der berechtigt ist, auf eine auf der Whitelist des Übungssystem-Kurses stehende Moodle-Kursumgebung (oder anderen Tool Consumer) zuzugreifen, auch das Recht hat, die Übungssystem-Aufgaben dieses Kurses per LTI (von besagtem Tool Consumer aus) zu starten. Die "normale" Autorisierungsprüfung des Online-Übungssystems für direkte Erstzugriffe (ohne LTI) ist damit standardmäßig außer Kraft gesetzt, die eigentliche Benutzer-Autorisierung wird komplett an den Tool-Consumer (in Abhängigkeit des LTI-Kontexts) delegiert. Dem liegt die Annahme zugrunde, dass jeder, der z.B. Teilnehmerrechte an einer Moodle-Kursumgebung hat, normalerweise auch alle darin eingebetteten Aufgaben aufrufen darf.

Daher wird, falls der Benutzer beim Übungssystem bei seinem Erstzugriff per LTI (nach Whitelist-Prüfung, s.o.) noch unbekannt ist, dieser beim Übungssystem-Kurs automatisch als "Beleger" (Teilnehmer) angemeldet (was nötig ist, damit ihm Einsendungen und Korrekturen zugeordnet werden können). Einzige Voraussetzung ist, dass ein gleichnamiger LDAP-Account existiert – andernfalls schlägt der Launch-Request fehl.

Dieser Mechanismus wird intern als Student-Auto-Enroll bezeichnet, in der Betreuer-UI nennt er sich »Studentenrechte vererben« (siehe oben). Der Betreuer kann diese »Studentenrechte vererben«-Option für jeden angemeldeten Tool-Consumer auf der Whitelist aber abschalten, dann gelten für alle Zugriffe – egal ob lokal oder per LTI – wieder dieselben Zugriffsregeln. In der Regel (nicht immer) läuft diese Übungssystem-seitige Autorisierungsprüfung dann auf die Prüfung einer Kursbelegung hinaus.

Tutor-Auto-Enroll oder lokale Autorisierung

Ähnlich verhält es sich mit Betreueraccounts: Nimmt ein Betreuer (des Tool Consumers, z.B. einer Moodle-Kursumgebung) einen LTI-Launch-Request vor, so wird nach erfolgreicher Whitelist-Autorisierung im Standardfall angenommen, dass ihm der Whitelist-Eintrag auch volle Betreuerrechte für den Übungssystem-Kurs, zu dem die Aufgabe gehört, gewährt. Es wird dann der Aufgabeneditor geöffnet, und das UI bietet auch die Funktion, ein eigenes Browserfenster mit vollständigem Betreuerzugang zu öffnen.

Anders als beim Studentenzugang ist dies jedoch kein *persistenter* Auto-Enroll, d.h. der jeweilige Nutzer bekommt die Betreuerrechte im Übungssystem nur temporär für die Dauer der Browsersitzung zugesprochen, ohne dass er permanent in die Liste der Kursbetreuer eingetragen wird. (Da den Betreueraccounts keine Daten wie Einsendungen, Korrekturen o.ä. zugeordnet werden, besteht dazu keine Notwendigkeit.)

Auch dieser Auto-Enroll-Mechanismus (im Beteuer-Ul entsprechend als »Betreuerrechte vererben« bezeichnet, s.o.), lässt sich zu jedem Whitelist-Eintrag einzeln abschalten. Ist er deaktiviert, können auch per LTI-Request nur solche Nutzer die Aufgabe bearbeiten oder andere Betreuertätigkeiten

07.03.25 Seite 26 / 57

durchführen, die im Ubungssystem-Kurs lokal als Betreuer angemeldet sind.

Session für Folge-Requests

Bei der Aufgabenbearbeitung kommt es zu Folge-Requests, z.B. Einsenden der Eingaben oder (nach Einsendeschluss) die Navigation von der Aufgabenansicht zur Korrektur oder Musterlösung. Hier werden ganz gewöhnliche, von LTI unabhängige HTTP-Requests vom Browser an das Übungssystem gesendet, und natürlich muss hier (da Links z.B. auch manipuliert werden könnten und ein Nutzer versuchen könnte, auch auf Inhalte anderer Kurse zuzugreifen) für jeden Request eine erneute Autorisierungsprüfung erfolgen. Auch muss sich das Übungssystem merken, welcher User die Requests sendet. Das ist nicht selbstverständlich, denn im "Normalbetrieb" arbeitet das Online-Übungssystem ohne Sessions nur nach HTTP-AUTH-Verfahren (aka "Basic Authentication"), bei dem der Browser einmalig Benutzername und Passwort vom User erfragt und bei jedem Folgerequest wieder mitsendet.

Speziell für LTI-Aufgaben wurden hier also Sessions im Übungssystem eingeführt und eine "kursspezifische Autorisierung": Beim LTI-Launch-Request wird (nach "Erst-Autorisierung") für den User eine neue Session geöffnet, und darin werden der eingeloggte User ebenso wie der aufgerufene Kurs gespeichert. In Folgerequests innerhalb derselben Session gilt der User als autorisiert, falls sich der Folgerequest auf denselben Kurs bezieht und für dieselbe Benutzerrolle zugänglich ist.

Launch-Requests

Aufrufvarianten (Launch-URL und Custom-Parameter)

Das Online-Übungssystem unterstützt inzwischen zwei Varianten eines Launch-Requests

- 1. Der Launch-URL identifiziert nicht nur das Online-Übungssystem und dort einen LTI-Launch-Request, sondern enthält auch bereits alle Angaben zur zu startenden Aufgabe bzw. zum zu öffnenden Aufgabenheft.
 - Beispiel für einen solchen Launch-URL: https://online-uebungssystem.fernuni-hagen.de/six/LTILaunch/01614/WS25/1/5 für die fünfte Aufgabe aus dem ersten Aufgabenheft der 01614-Umgebung aus Wintersemester 25/26 des Veranstalters six.
- 2. Der Launch-URL referenziert nur das System bzw. dessen LTI-Lauch-Request-Service und noch keine konkrete Aufgabe (bzw. Aufgabenheft). Um festzulegen, *was* genau geöffnet werden soll, müssen die nun im URL fehlenden Angaben statt dessen in "Custom LTI Parameters" übergeben werden.

Nummer 1 ist das ältere Verfahren, das wir bereits mit Moodle 2 und 3 benutzten, das aber ab Moodle 4.4 dort nicht mehr so einfach möglich ist (siehe Änderungen ab Moodle 4.4). Genauer kann man ab Moodle 4.4 nur noch in einer zentralen, für spätere Tool-Einbindungen zentral hinterlegten Moodle-Toolkonfiguration überhaupt einen URL angeben (zusammen mit den Zugangsdaten wie Key und Secret). Will man diese Zugangsdaten nicht zu jeder einzelnen Toolverknüpfung (Einbettung einer konkreten Aufgabe in eine konkrete Moodle-Lernumgebung) mit angeben, sondern den Lehrenden eben eine vom ZDI zentral bereitgestellte Toolkonfiguration zur Verfügung stellen, mit der sie beliebige Übungssystem-Aufgaben(hefte) in ihre Kursumgebungen einfügen können, dann müssen diese nun einen einheitlichen Tool-URL verwenden.

Daher wurde speziell für den Einsatz mit Moodle ab Version 4.4 das zweite Verfahren eingeführt.

07.03.25 Seite 27 / 57

Referenz auf eine Aufgabe bzw. ein Aufgabenheft

Der Primärschlüssel einer Aufgabe, also die Angaben, die eine konkrete Aufgabe im Online-Übungssystem eindeutig identifizieren, besteht aus folgenden Komponenten:

Schlüsselkomponente	Info	Beispiel
Veranstaltername	String, identifiziert einen Aufgabenanbieter im System	lgxy
Kursnummer	String, normalerweise fünfstelliger Ziffernstring	01614
Versionsnummer ⁶	Semesterkennung bestehend aus SS für Sommer- bzw. WS für Wintersemester, gefolgt von zwei Ziffern für das Jahr, in dem das Semester beginnt.	WS24
Aufgabenheftnummer	Zahl ≥ 1	1
Aufgabennummer	Zahl ≥ 1	5

Der Schlüssel eines Aufgabenhefts ist analog aufgebaut, nur entfällt dort die Aufgabennummer.

Launch-URL bei Variante 1 (vor Moodle 4.4)

Der Aufbau eines solchen Launch-Request-URLs, der obige Schlüsseldaten mit enthält, hat den folgenden Aufbau, wobei die kursiv gesetzten Teile durch die entsprechenden Schlüsselfelder (s.o.) zu ersetzen sind: https://online-uebungssystem.fernuni-

hagen.de/Veranstaltername/LTILaunch/Kursnr/Versionsnr/Aufgabenheftnr/Aufgabennr/. Bei Launch-URLs für ganze Hefte entfällt das letzte Token, d.h. der URL endet bereits hinter der Aufgabenheftnummer.

Vor Moodle 4.4 mussten Betreuer:innen diesen URL selbst in ihrer Lernumgebung bei der Tool-Einbindung eingeben – bzw. sie konnten ihn sich im Online-Übungssystem nach Auswahl einer Aufgabe bzw. eines Aufgabenhefts generieren lassen und per Copy&Paste in Moodle einfügen.

Launch-URL bei Variante 2 (ab Moodle 4.4)

Beim Verfahren Nr. 2 ist der Launch-URL konstant / aufgabenunabhängig. Er lautet nur noch kurz: https://online-uebungssystem.fernuni-hagen.de/LTILaunch⁷.

Im Falle von Moodle (ab V4.4) wird dieser Launch-URL nur einmalig in der zentralen Moodle-Toolkonfiguration (als dort sog, Tool-URL) hinterlegt, die Lehrenden kommen damit nicht mehr in Berührung. Diese müssen dann vielmehr den Schlüssel zur einzubettenden Aufgabe (bzw. Aufgabenheft) über Custom-Parameter festlegen.

Die Custom-Parameter bei Variante 2 (ab Moodle 4.4)

Ein LTI-Launch-Request ist ein gewöhnlicher HTTP-Post-Request, der eine Reihe von Parametern mitsendet, analog zum Einsenden eines HTML-Formulars mit Post-Method (Content-Type application/x-www-form-urlencoded). Der LTI-Standard und seine Extensions legen diverse Parameternamen fest. LTI-Custom-Parameter sind Post-Parameter, die mit dem Präfix custom_beginnen und ansonsten beliebig heißen dürfen. Diese werden vom aufzurufenden LTI-Tool festgelegt.

Das Online-Übungssystem erwartet bei einem Launch-Request nach Methode 2 (also mit dem oben

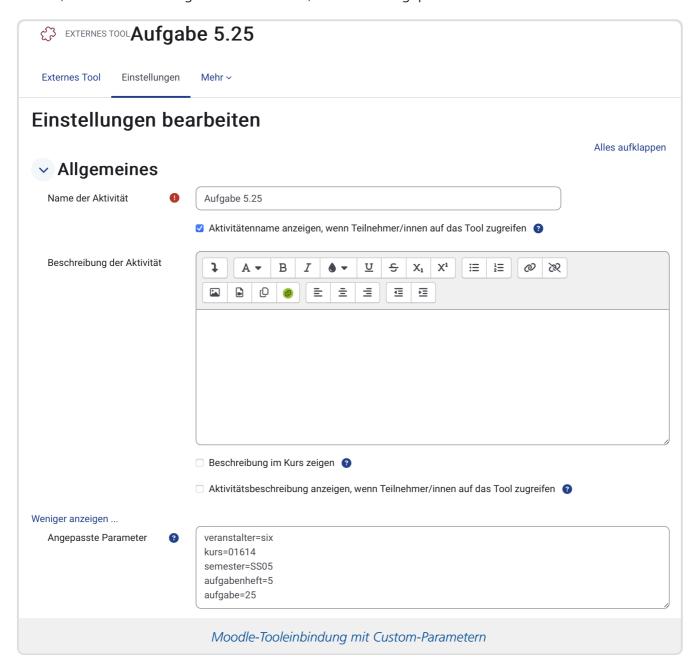
07.03.25 Seite 28 / 57

beschriebenen verkürzeten Launch-URL, der keinen Aufgaben(heft)-Schlüssel enthält) folgende Custom-Parameter, die eben einen Aufgabenheftschlüssel oder Aufgabenschlüssel enthalten:

Schlüsselkomponente	Custom-Parametername	Aliasse
Veranstaltername	veranstalter	veranstaltername
Kursnummer	kurs	kursnr
Versionsnummer	semester	version, versionsnr
Aufgabenheftnummer	aufgabenheft	aufgabenheftnr, heft
Aufgabennummer	aufgabe	aufgabennr

Den konkreten HTTP-Parametern ist dabei, wie gesagt, noch jeweils das Präfix custom_voranzustellen, also z.B. custom kurs.

Bei Einbettung eines externen Tools in Moodle 4.4 ff. erfolgt die Eintragung dieser Custom-Parameter in der (unter »Mehr anzeigen« zu findenden) Textarea »Angepasste Parameter«:



07.03.25 Seite 29 / 57

Die Syntax ist:

- Ein Parameter pro Zeile
- in der Form name=value,
- wobei name ein Parametername aus der obigen Tabelle ist (ohne custom -Präfix)
- und value der zugeordnete Wert.
- Die Reihenfolge beliebig.

Analog zu den Launch-URLs beim Verfahren 1 können auch hier die Betreuer:innen sich in ihrem Online-Übungssystem-Kurs nach Auswahl einer einzubindenden Aufgabe (oder eines Aufgabenhefts) einen solchen Parameter-String erzeugen lassen und per Zwischenablage in das Moodle-Eingabefeld kopieren.

Ablauf eines Launch-Requests

Dieser Abschnitt beschreibt zusammenfassend den Algorithmus der Launch-Request-Bearbeitung im Online-Übungssystem:

- Es wird zunächst geprüft, ob ein Lauch-URL *mit* Aufgaben(heft)-Schlüssel vorliegt. Falls nicht, werden die Custom-Parameter nach einem solchen Schlüssel durchsucht. Wurde kein Aufgabenheft oder keine Aufgabe vollständig spezifiziert, wird nur eine Fehlermeldung ausgegeben, ansonsten wie folgt fortgefahren.
- Es werden folgende Rollen unterstützt: Student (LTI-Rolle Learner) und Betreuer (LTI-Rollen Instructor, ContentDeveloper oder TeachingAssistant diese drei Rollen werden von der ToolProvider-Library als "Staff" zusammengefasst und erlauben Betreuerzugriff auf den Übungssystem-Kurs). Andernfalls wird ein Zugriff verweigert.⁸
- Eine ggf. bereits bestehende Session wird invalidiert und eine neue gestartet.
- Es wird für die Session der (I)Frame-Modus aktiviert, falls der Launch-Request einen IFrameoder Frame-Modus im launch_presentation_document_target-Parameter
 angefordert hat. Wird das Tool jedoch in einem neuen Popup-Fenster geöffnet, so bleibt das
 "normale", vollständige Übungssystem-Seitenlayout aktiv. Diese Entscheidung wird in der
 Session festgehalten. (Details dazu finden sich auch im Abschnitt Return-Links nur im (I)FrameModus)
- Der Return URL (LTI-Parameter launch_presentation_return_url) wird in der Session hinterlegt, um später entsprechende Zurück-Links einblenden zu können. (Siehe auch: LTI-Return-URL)
- Anhand der Tool-Consumer-Einstellung »Username Parameter« (siehe Tool-Consumer-Registrierung) wird im Launch Request nach dem Loginnamen des Users gesucht. Dies ist für die Übungssystem-Implementierung ein Pflicht-Parameter (auch wenn im LTI-Standard weder verpflichtend noch genormt), d.h. ohne diese Angabe kann keine Autorisierung erfolgen und der Zugriff wird verweigert. (Siehe auch Datenschutz-Einstellungen zum Aktivieren der Übermittlung dieses Usernamens in Moodle.)
- Im Folgenden bezeichne username den besagten im Launch-Request mitgesendeten Loginnamen des von Moodle bereits authentifizierten Nutzers, dessen Autorisierung fürs Übungssystem nun noch zu prüfen ist.
- Autorisierungsprüfung beim Launch-Request: Suche in Datenbank nach einem Whitelist-Eintrag zum LTI-Kontext des Aufrufs.
 - Existiert *kein* solcher Eintrag?
 - Bei Aufruf in Studentenrolle: Lehne Tool-Zugriff mit Fehlermeldung ab!

07.03.25 Seite 30 / 57

- Bei Aufruf in Betreuerrolle: Redirect zu einer Webseite mit Möglichkeit zum Whitelisting (mit lokalem Betreuer-Login), wie oben abgebildet.
- Sonst (Whitelist-Eintrag existiert):
 - Bei Aufruf in Studentenrolle: Damit Einsendungen des Nutzers im Übungssystem gespeichert werden können, muss Übungssystem-lokal ein entsprechender Benutzeraccount mit Belegung des Übungssystem-Kurses existieren. Ist das (noch) nicht der Fall, wird der lokale Account ggf. automatisch neu erzeugt und zum Kurs angemeldet.
 - Falls im Whitelist-Eintrag die Option »Studentenrechte vererben« (Auto-Enroll) eingeschaltet ist (Default), muss dazu lediglich ein zentraler LDAP-Account vorliegen⁹, andernfalls wird der Zugriff verweigert.
 - Ist das Auto-Enrollment abgeschaltet, erfolgt zunächst eine reguläre Studenten-Autorisierungsprüfung. In der Regel wird dazu geprüft, ob der Student den Kurs laut SLO/BGS belegt hat (in Kursparametern abschaltbar), in Sonderfällen kann für bestimmte Veranstalter auch eine abweichende Autorisierung eingerichtet sein, z.B. dass nur Studenten berechtigt sind, die in einen bestimmten Studiengang eingeschrieben sind. Stellt das Übungssystem keine Autorisierung fest, wird der LTI-Zugriff verweigert.
 - Die Übungssystem-lokale Unterrolle des neu zu erzeugenden Studentenzugangs wird vom Loginnamen abhängig gemacht: Echte FernUni-Studenten (¬+Matrikelnr) werden als STUDENT, Mentoren (¬+Beziehernr) als MENTOR, andere User (insb. z.B. Moodle-Betreuer, die per »Rolle wechseln« in Moodle die Teilnehmerrolle annehmen, um die Aufgabe zu testen) werden als TESTSTUDENT angemeldet.
 - Nach erfolgreicher Prüfung der Autorisierung des Studenten wird außerdem sofern noch nicht vorhanden ein Tripel erzeugt und persistiert, welches das von der LTI-Tool-Provider-Library automatisch angelegte User-Objekt mit der Aufgabe bzw. dem Aufgabenheft und dem Studenten verknüpft und später vom Outcomes-Service genutzt werden kann, um Korrekturpunkte ins Moodle-Gradebook einzutragen. Je nachdem, ob ein Aufgabenheft oder eine einzelne Aufgabe als Tool eingebunden wurde (ob also der Launch-URL am Ende eine Aufgabennummer enthält oder nicht), wird entweder ein Tripel der Klasse StudentAufgabenheftLtiUserTripel oder StudentAufgabenheftLtiUserTripel erzeugt (siehe Klassendiagramm).
 - Dieser Schritt findet jedoch nur statt, falls der LTI-Request den Parameter lis_result_sourcedid enthielt, welcher genau dazu dient, dem Toolprovider die Rückübermittlung eines Results über den Outcomes-Service zu ermöglichen. In Moodle gilt: Falls in den Datenschutz-Einstellungen das Akzeptieren von Ergebnisübertragungen abgeschaltet ist, also Outcomes nicht genutzt werden kann, wird auch dieser Parameter nicht übertragen, und das Online-Übungssystem wird diese Datensätze nicht anlegen. Der Outcomes-Dienst im Online-Übungssystem wird dann nicht funktionieren, selbst wenn er hier eingeschaltet wird.
 - Abschließend wird die Aufgabenseite bzw., die Aufgabenheftübersicht als HTML-Antwort zur Anzeige im IFrame oder im neuen Fenster (s.o.) ausgeliefert.

07.03.25 Seite 31 / 57

Bei Aufruf in Betreuerrolle:

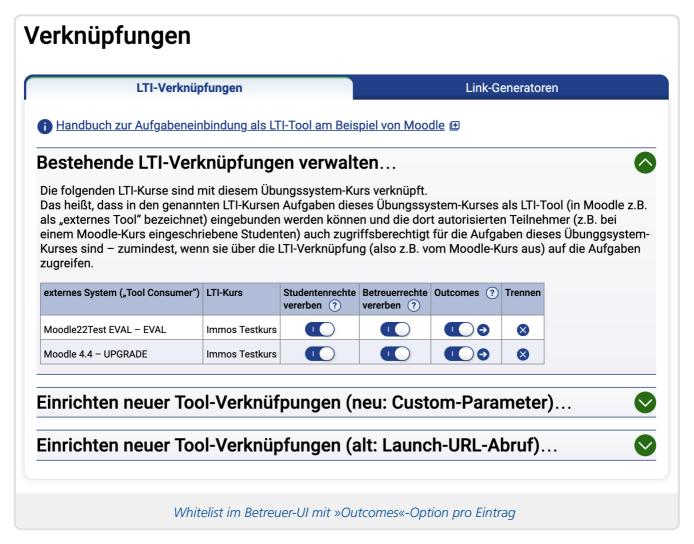
- Ist die Option »Betreuerrechte vererben« für den LTI-Kontext abgeschaltet und ist im Übungssystem-Kurs lokal kein Betreueraccount angemeldet, dessen Loginname mit username übereinstimmt, dann lehne den Zugriff mit Fehlermeldung ab.
- Sonst erlaube den Betreuerzugriff: Da Betreuer nur globale Daten im Kurs ändern und keine Betreueraccount-spezifischen Daten wie Einsendungen oder Korrekturen besitzen, wird anders als bei Studentenrolle kein lokaler Account für Betreuer benötigt/angelegt. Bei vorliegender Kurs-Autorisierung (Whitelist-Eintrag) kann sich jeder, der bei der einbindenden Moodle-Kursumgebung Betreuungsrechte (Staff, s.o.) hat, auch per LTI als Betreuer im Übungssystem anmelden, ohne im Übungssystem selbst als Betreuer des Kurses erfasst sein zu müssen.
- Abschließend wird bei einer eingebetteten Aufgabe dem Betreuer der Aufgabeneditor, bei eingebettetem Aufgabenheft die Aufgabenheftverwaltung als anzuzeigender Tool-Inhalt ausgeliefert.
- Für einen so authentifizierten und autorisierten Benutzer wird eine spezielle *kursspezifische Autorisierung* in der Session abgelegt, d.h. das Übungssystem erlaubt in derselben Session alle Folgerequests, die sich auf denselben Kurs beziehen und für die Benutzerrolle (Betreuer oder Student) zugänglich sind.
 - Versucht z.B. ein LTI-Learner, auf Betreuer- oder Korrektor-Services zuzugreifen, wird der Zugriff verweigert, da der User nicht die richtige Rolle hat.
 - Versucht z.B. der LTI-Betreuer, auf Betreuerseiten eines anderen Kurses zuzugreifen, wird der Zugriff verweigert, da die Session nur zum Zugriff auf Inhalte des im Launch-Request referenzierten Kurses autorisiert.
 - In all solchen Fällen enthält die Fehlerseite einen Link, der ein Beenden der Session und lokalen Login ermöglicht.

07.03.25 Seite 32 / 57

Outcomes: Aufgabenergebnisse an Moodle übermitteln

Neben dem zentralen Feature des Launch-Requests, dem sich das vorangehende Kapitel widmete, umfasst der LTI-Standard inzwischen weitere optionale Features, die auf Tool-Consumer-Seite in Form von WebServices angeboten werden, welche durch Tool-Provider aufgerufen werden können. Viele wurden erst mit LTI 2 eingeführt und kommen hier nicht zur Anwendung. Wie oben aber bereits angekündigt, nutzt das Online-Übungssystem den mit LTI 1.1 offiziell eingeführten Outcomes-Service, um Aufgabenbewertungen an Moodle zu übertragen.

Dabei ist die Verwendung des Outcomes-Services optional: Der Kursbetreuer legt für jeden Whitelist-Eintrag einzeln fest, ob für diese Verknüpfung der Outcomes-Service aktiviert werden soll oder nicht, vgl. folgende Abbildung:



Eine weitere Voraussetzung ist, dass auch in Moodle das Akzeptieren von Ergebnis-Übermittlungen aktiviert wurde, siehe Datenschutz-Einstellungen.

Welche Ergebnisse können übertragen werden (und wann)?

Hier sind zunächst folgende Besonderheiten des Online-Übungssystems zu beachten:

• Wenn ein Student zu einer Aufgabe etwas einsendet, liegt zunächst einmal lediglich eine Einsendung vor und ihm wird eine Quittungsseite angezeigt. Es liegt noch keine Bewertung vor

07.03.25 Seite 33 / 57

- selbst wenn, wie bei Selbsttestaufgaben, die Quittung eine Vorkorrektur enthalten kann, die wiederum eine Bewertung vorwegnehmen kann. Die Aufgabe ist aber noch zur Bearbeitung offen und aus Übungssystem-Sicht existiert noch keine bewertete Korrektur.
- Nach endgültiger Abgabe ("Heft-Schließen") kann ein Student seine Einsendungen nicht mehr verändern, und das Übungssystem generiert eine Korrektur zu den Einsendungen eines Studenten zu einer Aufgabe. Bei aktivierter Autokorrektur kann diese bereits fertig »korrigiert« sein, aber Aufgaben können auch manuell korrigiert werden, im welchem Fall die "Korrektur" zunächst im Status »unkorrigiert« vorliegt, also streng genommen ein erst noch (von einem menschlichen Korrektor) zu korrigierendes Dokument darstellt. Der Korrektor kann die Korrektur annotieren, bewerten (d.h. ihr eine erreichte Punktzahl zuordnen) und sie schließlich in den Status »korrigiert« versetzen.
- Ein Betreuer gibt später die »korrigierten« Korrekturen frei, wodurch sie in den Zustand »freigegeben« übergehen und dem Studenten zur Einsicht angeboten werden. Bei Autokorrektur kann die Freigabe optional sofort beim Heft-Schließen automatisch erfolgen.

Erst für »freigegebene« Korrekturen steht im Übungssystem ein Aufgabenergebnis bereit, das ans Moodle-Gradebook übertragen werden kann: Jede fertige Korrektur enthält eine erreichte Punktzahl (vom Betreuer oder Korrekturmodul vergeben) und ist einer Aufgabe zugeordnet, zu welcher wiederum eine maximal erreichbare Punktzahl gespeichert ist. Daraus kann (bei mehr als 0 erreichbaren Punkten) eine Prozentpunktzahl berechnet werden, die wiederum per Outcomes-Service übertragen werden kann – sofern die einzelne Aufgabe als LTI-Tool eingebunden wurde. Falls ein ganzes Aufgabenheft als Tool eingebunden wird, so kann ebenfalls nur eine Prozentpunktzahl für pro Benutzer und Tool als "Outcome" übertragen werden. In diesem Fall errechnet sich das Ergebnis als Prozent des Heft-Gesamtergebnisses (Summe aller Aufgabenpunkte eines Teilnehmers dividiert durch Summe der in allen Aufgaben des Hefts maximal erreichbaren Punkte).

Aufgaben mit 0 erreichbaren Punkten (unbewertete Aufgaben oder reine Selbsttestaufgaben, s.u.) werden vom Outcomes-Dienst grundsätzlich ausgeschlossen, da es zu ihnen keine übermittelbaren Prozentpunkte geben kann.

Der Aufruf des Outcomes-Dienstes erfolgt (zumindest im ersten Versuch, siehe: Optimistische Service-Calls) unmittelbar bei der Freigabe einer Korrektur. Der WebService ist für jede zu übermittelnde Punktzahl einzeln aufzurufen, ermöglicht also keinen Bulk-Transfer von mehreren Ergebnissen.

Im Falle eines als Tool eingebundenen ganzen Aufgabenhefts wird ein Outcome-Ergebnis erst übertragen, sobald *alle* Korrekturen des Studenten zum Aufgabenheft korrigiert und freigegeben sind, das Gesamtergebnis also feststeht. Es werden also bei noch unvollständigen Korrekturen keine Teilergebnisse als Outcome übermittelt.

Außerdem erfolgt im Fall, dass die Freigabe einer Korrektur zurückgezogen oder die Korrektur (im Rahmen eines Heft-Öffnens) wieder gelöscht wird, ein erneuter Aufruf des Outcomes-Services, und zwar zum Löschen des korrespondierenden Gradebook-Eintrags (zur Aufgabe bzw. zum die Aufgabe enthaltenden Aufgabenheft).

07.03.25 Seite 34 / 57

Spezialfall Selbsttestaufgaben

Reine Selbsttestaufgaben, die sofort bei Einsendung eine Bewertung der Eingaben (als Vorkorrektur im Rahmen der Quittung) anzeigen, werden Übungssystem-intern normalerweise als unbewertete Aufgaben behandelt, d.h. der Aufgabe sind 0 erreichbare Gesamtpunkte zugeordnet und die Korrekturen, die nach Heft-Schließen erzeugt werden, werden ebenfalls mit 0 Punkten bewertet (unabhängig von einer ggf. im Text der Autokorrektur enthaltenen Bewertung). Der Grund ist, dass in solchen Selbsttestaufgaben in der Regel (wenn die Vorkorrektur die Musterlösung verrät) jeder Teilnehmer mühelos bei der zweiten Einsendung die volle Punktzahl erreichen kann und diese Aufgaben daher keine Ergebnisse liefern sollen, die sich mit "echten", erst nach Einsendeschluss/Heft-Schließen bewerteten Einsendearbeiten vermischen, also den Schnitt aufwerten. Solche Selbsttestaufgaben können ihre Ergebnisse auch nicht ins Moodle-Gradebook übermitteln. (Konkret werden, wie gesagt, alle Aufgaben mit 0 erreichbaren Punkten vom Outcomes-Dienst ausgeschlossen.)

Man kann von dieser Regel aber abweichen und auch Selbsttestaufgaben ins Gesamtergebnis einfließen lassen. In diesem Fall stehen also Punkte bereit, und diese lassen sich auch per Outcomes-Service an Moodle übermitteln – eine solche Übertragung ins Gradebook erfolgt dann aber *nicht* (wie die Sofortbewertung) unmittelbar nach Einsendung, sondern erst nach dem Heft-Schließen, wenn die bewertete Autokorrektur erstellt und freigegeben wird.

Technische Realisierung der Outcomes-Serviceaufrufe

Der eigentliche Service-Aufruf (einfacher HTTP-Request mit XML-Payload¹⁰, aber nicht nach SOAP-Standard) wird von der eingangs beschriebenen Library implementiert, d.h. das Übungssystem muss "nur" eine entsprechende Library-Funktion ausführen. Higher-Level-Methoden zum Löschen oder Updaten von Gradebook-Einträgen über den Outcomes-Service, welche die Nutzung der Library verkapseln, wurden im Übungssystem in einer eigenen Klasse LTIServices im Paket WebAssign.externalSystems implementiert.

Die Library-Routine zum Outcomes-Service erwartet im Wesentlichen (neben dem Modus "Write" oder "Delete") zwei Argumente: Ein Outcome -Objekt und ein User -Objekt, wobei es sich um Klassen aus dieser Library handelt. Eine User -Instanz beschreibt – anders als ihr Name vermuten lässt – nicht etwa nur den Benutzer (Student), dessen Ergebnis zu aktualisieren ist, sondern ein Paar aus User und Resource-Link, wobei der Resource-Link wiederum die LTI-seitige Aufgabenressource (External-Tool-Ressource) referenziert. D.h. das User -Objekt referenziert tatsächlich exakt einen Gradebook-Eintrag eines Teilnehmers zu einer Aufgabe. Das Outcome -Objekt wiederum enthält die Prozentpunktzahl, die unter jedem Gradebook-Eintrag einzutragen ist (außer bei Löschoperation).

Die Übungssystem-Datenbank enthält eigene Datenbanktabellen, die der LTI-Toolprovider-Library zur Persistierung ihrer Entitätsklassen dienen. Die Daten des <code>User</code>-Objekts werden hier aus den Tabellen <code>lti_user</code> (enthält User-ID und Foreign Key des Resource-Links), <code>lti_context</code> (enthält Daten des Resource Links und Foreign Key des Tool-Consumers) und <code>lti_consumer</code> (enthält Daten zum Consumer) geladen.

(Wie schon oben in Ablauf eines Launch-Requests beschrieben, werden diese <code>User</code>-Instanzen (und ebenso die im Folgenden beschriebenen Tripel dazu) nur angelegt, wenn der Tool-Consumer überhaupt Result-Messages akzeptiert und dazu beim Launch-Request den Parameter <code>lis result sourcedid</code> mitsendet.)

07.03.25 Seite 35 / 57

Erste Tripelklasse zur Assoziation von Gradebook-Entries zu Korrekturen

Dieser Abschnitt behandelt Tripel für den Fall einer einzeln als Tool eingebundenen Aufgabe.

Damit das Online-Übungssystem zu einer vorliegenden (z.B. gerade freigegebenen) Korrektur den Outcomes-Dienst aufrufen kann, muss es – wie vorangehend beschrieben – zunächst eine User-Instanz ermitteln, welche den zu aktualisierenden Gradebook-Eintrag referenziert.

Dazu musste im Übungssystem-Model (Entitätsklassen und Datenbanktabellen) eine (persistente) Assoziation zum Lookup solcher User-Instanzen zu einer Korrektur geschaffen werden. Realisiert wurde das in Form einer Entitäsklasse StudentAufgabeLtiUserTripel, persistiert in Tabelle Student__Aufgabe__lti_user und im Folgenden kurz als "die Tripelklasse" bezeichnet. Ein Tripel (Instanz der Tripelklasse) verknüpft genau einen Studenten, eine Aufgabe und ein LTI-User-Object (das, wie gesagt, einen Gradebook-Eintrag identifiziert). Da eine Korrektur wiederum stets zu genau einer Aufgabe und einem Studenten gehört (in UML als Assoziationsklasse darstellbar), lässt sich zu jedem Paar (Korrektur, User) auch genau ein Tripel (Aufgabe, Student, User) finden.

Dass kein Korrektur-User-Paar, sondern ein Aufgaben-Student-User-Tripel angelegt wird, liegt darin begründet, dass die Tripel bereits zu einem Zeitpunkt erzeugt werden, an dem die Korrekturen noch gar nicht existieren, nämlich im Rahmen eines LTI-Launch-Requests.

Hinweis: Die Anzahl der Tripel zu einer Korrektur und damit die Abbildung von einer Korrektur auf einen LTI-User sind *nicht eindeutig*:

- Zum Einen könnte es Korrekturen ohne zugehörige Gradebook-Einträge und somit ohne zugehörige Tripel geben, nämlich, wenn die Einsendungen zur Korrektur gar nicht über LTI erfolgten, sondern die Aufgabe regulär lokal im Online-Übungssystem bearbeitet wurde.
- Zum Anderen könnte es *mehrere Tripel zu einer Korrektur* geben, auch wenn das ein eher theoretischer Sonderfall ist, der kaum praktische Relevanz haben dürfte: Dieselbe Übungssystem-Aufgabe könnte in *mehreren* Moodle-Ressourcen verlinkt worden sein. Wenn dann auch noch derselbe Student auf mehrere dieser Moodle-Ressourcen Zugriff hat und diesen auch ausnutzt, also dieselbe Übungssystem-Aufgabe aus verschiedenen Moodle-Ressourcen heraus startet, dann (und nur dann) wird für jede davon auch ein Tripel angelegt und das Aufgabenergebnis entsprechend über den Outcomes-Service in die Gradebooks aller Moodle-Ressourcen eingetragen, von denen aus der Student die Aufgabe einmal gestartet hat.

Zweite Tripelklasse zur Assoziation von Gradebook-Entries zu Aufgabenheft-Leistungen

Falls ein ganzes Aufgabenheft (statt einer einzelnen Aufgabe) als Tool eingebunden wird, wird eine zweite Tripelklasse verwendet.

In diesem Fall soll ja nicht das Ergebnis einer einzelnen Korrektur als Outcome übertragen werden, sondern das Gesamtergebnis zu einem Aufgabenheft. Dieses wird im Model durch die Assoziationsklasse AufgabenheftLeistung dargestellt, wobei es sich um eine rein transiente Klasse ohne korrespondierende Datenbanktabelle handelt, da das Heftergebnis immer aus den Einzelergebnissen (Korrekturen) abgeleitet, also neu berechnet werden kann.

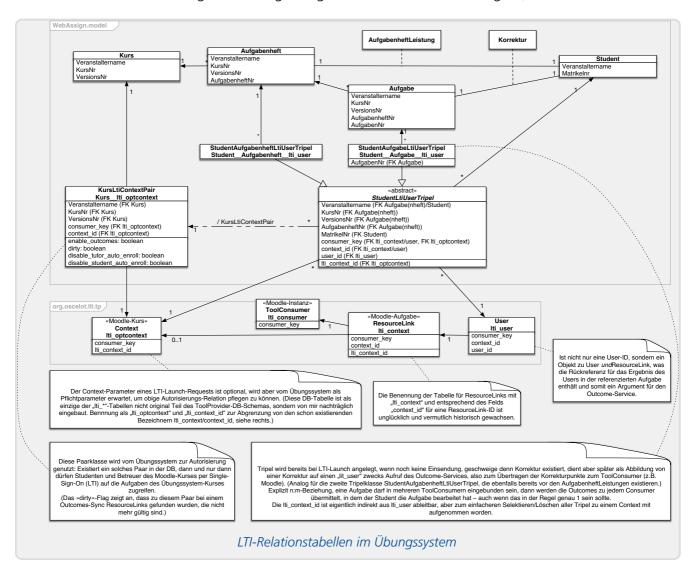
Auch hier gilt aber, dass das Tripel schon angelegt wird, bevor überhaupt Einsendungen, geschweige denn Korrekturen oder ein Gesamtergebnis vorliegen, d.h. es wird ein Tripel aus Aufgabenheft (statt

07.03.25 Seite 36 / 57

Aufgabe) sowie Student und LTI-User-Objekt erzeugt.

Effektiv unterscheiden sich die beiden Tripelklassen nur darin, dass statt einer Referenz auf eine Aufgabe eine Referenz auf ein Aufgabenheft enthalten ist. In den Java-Klassen wurde daher eine abstrakte Superklasse beider Tripel modelliert, deren Subklassen nur die jeweilige Referenz hinzufügen. Auf Datenbankebene gibt es je eine Tabelle pro Tripelklasse, die sich beide nur darin unterscheiden dass Student_Aufgabe_lti_user eine zusätzliche Spalte AufgabenNr enthält.

Zur Übersicht sei hier nochmals das Klassendiagramm wiederholt, das die beteiligten Datenbanktabellen mit ihren Schlüssel- und ausgewählten Nichtschlüsselattributen auflistet. (Die Relationen lassen sich analog auf die zugehörigen Entitätsklassen übertragen.)



LTI-Only-Mode

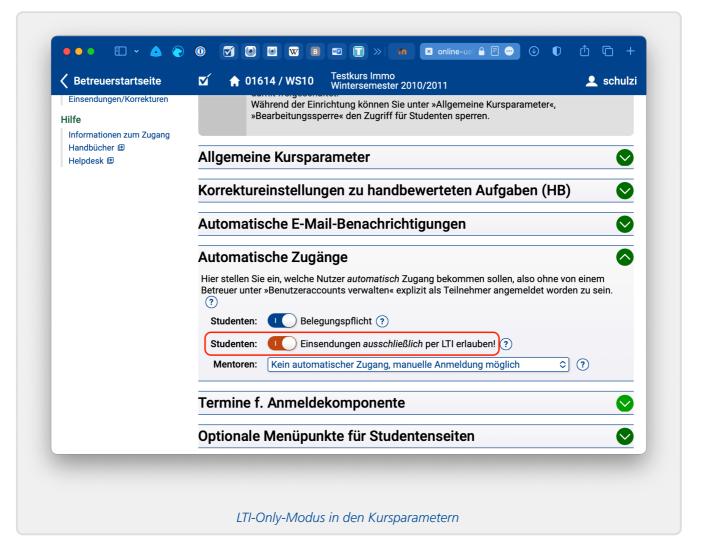
Im vorhergehenden Abschnitt wurde erläutert, dass bei einem LTI-Launch ein Tripel gespeichert wird, welches benötigt wird, um später per Outcomes-Service die Korrekturpunkte ins Moodle-Gradebook zurückzuschreiben.

Das heißt aber auch: Wenn ein Student eine Aufgabe zwar von Moodle aus per LTI starten könnte, es aber nicht tut, sondern sich zur Aufgabenbearbeitung direkt beim Online-Übungssystem anmeldet und die Aufgabe dort lokal startet und ausführt, dann kann sein Aufgabenergebnis nicht in seine Moodle-Umgebung übertragen werden.

07.03.25 Seite 37 / 57

Aus diesem Grund kann ein Kursbetreuer, der seine Übungssystem-Aufgaben alle über Moodle anbieten und dabei auch den Outcomes-Service nutzen möchte, einen *lokalen Erstzugriff* auf eine Aufgabe im Übungssystem verbieten. Ein Student *muss* dann eine Aufgabe (bzw. die Übersichtsseite eines Aufgabenhefts) zumindest einmal von Moodle aus starten, so dass das jeweilige Tripel erzeugt wird. (Existiert dieses, kann der Student anschließend die verknüpfte Aufgabe bzw. alle Aufgaben des verknüpften Hefts auch Übungssystem-lokal öffnen.)

Diese Einstellung wird intern kurz als "LTI-Only-Mode" bezeichnet. Ein Kursbetreuer findet dazu in den Kursparametern eine entsprechende Checkbox »Einsendungen *ausschließlich* per LTI erlauben«. Diese wird nur angeboten, wenn LTI überhaupt in Verwendung ist, genauer: wenn die Whitelist nicht leer ist.



Der LTI-Only-Modus wirkt global für den ganzen Kurs und bedingt, dass alle Aufgaben in diesem Übungssystem-Kurs ausschließlich zur LTI-Einbindung vorgesehen sind, denn eine lokale Aufgabenbearbeitung ist dann nicht mehr möglich.

Eine technische Einschränkung sei jedoch nicht unerwähnt: Der gesamte Mechanismus ist darauf ausgelegt, dass eine Aufgabe bzw. ihr Aufgabenheft nur *einmal* als LTI-Tool in eine externe Lernumgebung eingebunden wird. Falls es *mehrere* LTI-Einbindungen einer Aufgabe als Tool gibt (direkt als Einzelaufgaben-Tool oder auch indirekt als Aufgabenheft-Tool), genügt es, über *eines* dieser LTI-Tools die Aufgabe bzw. eine Aufgabe ihres Hefts zu starten, um die LTI-Only-Sperre aufzuheben. Startet ein User die Aufgabe dann einmal über Tool 1, ist anschließend auch ein Direktzugriff auf die Aufgabe möglich, ohne dass sichergestellt wird, dass z.B. auch zu Tool 2 ein Gradebook-Eintrag erzeugt wird.

07.03.25 Seite 38 / 57

Optimistische Service-Calls und Sync-Jobs

Prinzipiell erfolgen die Aufrufe des Outcomes-Services "live", d.h. direkt bei einem der folgenden Events:

- Freigabe einer Korrektur (→ Outcome speichern)
 - Automatische Freigabe einer Autokorrektur direkt im Rahmen des Heftschließens
 - Manuelle Freigabe einer Korrektur durch einen Betreuer (bei manuellen Korrekturen oder bei Autokorrekturen, wenn für diese keine automatische Freigabe eingeschaltet war)
- Zurücksetzen des Freigabestatus einer Korrektur durch den Betreuer (→ Outcome löschen)
- Heft-Öffnen durch Betreuer, wodurch alle Korrekturen zum Heft gelöscht werden (→ Outcomes löschen)

Anmerkung: Falls ein ganzes Aufgabenheft statt einer einzelnen Aufgabe als LTI-Tool eingebunden wurde, werden dennoch dieselben Ereignisse als Träger verwendet: Sobald eine Korrektur freigegeben wurde, wird überprüft, ob damit nun alle Korrekturen des Studenten zum Heft freigegeben sind und somit das Ergebnis feststeht, in welchem Fall ein Outcome gespeichert wird. Und falls eine Freigabe zurückgezogen oder das Heft geöffnet wird, wird das Outcome wieder gelöscht.

Nun ist es bei jeglicher Netzwerkaktivität zwischen zwei Servern natürlich möglich, dass Fehler auftreten. Z.B. könnte Moodle gerade nicht erreichbar sein oder das LAN gestört sein. Es gibt prinzipiell zwei Möglichkeiten, in obigen Vorgängen auf solche Fehler zu reagieren:

- 1. "Transaktional": Ein Vorgang soll atomar sein, im Fehlerfall soll die Transaktion gar nicht durchgeführt werden. Wenn z.B. im Rahmen einer Korrekturfreigabe der Outcome-Service nicht fehlerfrei ausgeführt werden kann, soll auch die Freigabe selbst nicht erfolgen oder sofort zurückgenommen werden.
- 2. "Optimistisch": Das Übungssystem versucht, die Services aufzurufen und hofft, dass das klappt. Im Fehlerfall soll aber die "Primäraktion" des Übungssystems dadurch nicht behindert werden und der Outcome-Service später nachgeholt werden. Wenn also z.B. im Rahmen einer Korrekturfreigabe der Outcome-Service nicht funktioniert, soll die Korrektur dennoch sofort freigegeben werden, nur der Gradebook-Eintrag auf Moodle-Seite fehlt dann vorerst und wird später nachgetragen.

Die zweite, "optimistische" Variante wurde implementiert, um primäre Aktionen wie Korrekturfreigaben nicht durch LTI-Probleme auszubremsen, die Gradebook-Einträge wurden als sekundär eingestuft.

Das bedeutet andererseits natürlich, dass ein Mechanismus für die Nachholung solcher Outcomes-Aktionen geschaffen werden musste. Dafür wurde ein "Sync-Job" implementiert, also ein Verfahren zum Abgleich zwischen den Korrekturen eines Kurses und den damit ggf. assoziierten LTI-Gradebook-Einträgen. Der Sync-Job eignet sich dabei nicht nur zum Nachholen fehlgeschlagener Service-Calls, sondern ermöglicht obendrein auch noch ein nachträgliches Einschalten des Outcomes-Features mit nachträglicher Übermittlung bereits vorliegener Korrekturpunkte.

Dieser Sync-Job kann manuell gestartet oder regelmäßig (z.B. täglich) als sog. Cronjob ausgeführt werden. Dazu kann er gezielt für einen einzelnen Whitelist-Eintrag (Tabelle Kurs_lti_optcontext) aufgerufen werden (vorgesehen für manuellen Aufruf) oder z.B. für alle Kurse eines Semesters (vorgesehen für Cronjob). In letzterem Fall kann wahlweise ein Semester explizit angegeben werden oder eine automatische Semesterwahl verwendet werden. Letztere führt einen Abgleich für alle Kurse des gerade laufenden Semesters durch und in einer einstellbaren

07.03.25 Seite 39 / 57

Zeitspanne (derzeit 15 Tage) nach einem Semesterende auch noch zusätzlich für das zurückliegende Semester.

Der Job kann als "echter Sync-Job" gestartet werden oder "im Löschmodus" – letzteres vorgesehen für den Fall, dass ein Betreuer die Outcomes-Option zu einem Whitelist-Eintrag wieder abschaltet, woraufhin alle ggf. schon übertragenen Outcomes-Einträge wieder zurückgenommen werden.

Aufgerufen für eine Liste von Kursen bestimmt der Sync-Job zunächst die Menge aller Whitelist-Einträge (Kurs__lti_optcontext) zu diesen Kursen, für die das enable_outcomes -Flag gesetzt ist (außer beim "Lösch-Job", der nach dem Löschen dieses Flags für einen Whiteliste-Eintrag aufgerufen wird). Die Liste kann ggf. anhand von weiteren Parametern gefiltert werden. Dann werden für jeden verbliebenen Whitelist-Eintrag die zugehörigen Tripel geladen und für diese der Outcomes-Service aufgerufen.

Sync-and-Clean-Job

Beim Outcomes-Service-Aufruf kann eine bestimmte Fehlersituation eintreten, nämlich, dass der zum übergebenen User-Objekt assoziierte Gradebook-Eintrag gar nicht existiert – weil auf Consumerseite der entsprechende User oder die Ressource (Aufgabe) nicht (mehr) existiert.

Gerade in Testumgebungen könnte auf Moodle-Seite eine »Externes Tool«-Ressource eingefügt und mit einer Übungssystem-Aufgabe verknüpft und damit auch eine Testeinsendung vorgenommen werden, und anschließend könnte diese Ressource in Moodle wieder gelöscht werden. Das zugehörige <code>User</code>-Objekt (Datenbankeintrag in <code>lti_user</code>) und ein dazugehöriges Tripel existieren dann im Übungssystem weiter und werden vom Sync-Job weiterhin bei jedem Abgleich bearbeitet. Moodle wird dann beim Outcomes-Aufruf jedes Mal einen entsprechenden Fehler melden.

Die Tool-Provider-Library sieht in ihrer Originalfassung leider kein brauchbares Fehlerhandling vor, mit dem sich diese Situation erkennen ließe: Die Service-Funktion liefert dort immer nur ein Boolean-Flag (erfolgreich ja/nein?), das keine Rückschlüsse auf die Art des Fehlers (z.B. Moodle nur vorübergehend nicht erreichbar oder eben auch Ressource endgültig gelöscht) zulässt. Dabei ist diese Unterscheidung für den Sync-Job relevant: Bei temporären Fehlern soll der Zugriff später erneut probiert werden, bei finalen Fehlern dagegen gibt eine (aussichtslose) regelmäßige Wiederholung keinen Sinn. Daher wurde der Übungssystem-interne Fork der Library entsprechend ergänzt, um diese Situation behandeln zu können:

Um den praktisch ganz ohne Exception Handling arbeitenden Library-Code minimalinvasiv zu erweitern, wurde Library-intern nur eine neue Property hinzugefügt, mit der sich nach einem Service-Aufruf der Statuscode der HTTP-Response abfragen lässt – sofern überhaupt eine Antwort empfangen wurde. Meldet ein Moodle-Server bei Aufruf des Outcome-Services per HTTP-Request an einen entsprechenden URL einen 404-Code (NOT FOUND), so ist das ein Signal, dass die angeforderte Ressource nicht mehr existiert. (Das ist offenbar in der – ohnehin eher "schwammigen" – LTI-1.1-Spezifikation https://www.imsglobal.org/specs/ltiv1p1/implementation-guide nicht geregelt, aber die Arbeit mit solchen Status-Codes ist bei derartigen HTTP-Post-Requests, insb. bei REST-Services, üblich. Dennoch kann ich nicht garantieren, dass andere Tool-Consumer als Moodle ebenfalls diesen Code senden!)

07.03.25 Seite 40 / 57

UPDATE: Moodle 2 sandte bei fehlendem Resource Link tatsächlich noch einen 404-Code, ab Version 3 funktioniert das in Moodle leider nicht mehr: Es wird immer mit Code 200 geantwortet. Damit funktioniert dieser Mechanismus leider nicht mehr. Die im Online-Übungssystem implementierte Fallunterscheidung wurde jedoch beibehalten, um zumindest im Fall, dass mal ein anderer Tool Consumer als Moodle 3 eingesetzt werden sollte, der doch diese Response-Code-Unterscheidung vornehmen sollte, auch sprechendere Fehlermeldungen ermöglicht werden.

Die von der Library abstrahierende Übungssystem-Klasse LTIServices wiederum rüstet dann ein Exception-Handling nach: Sie wirft immer, wenn ein Serviceaufruf fehlschlug, eine (Übungssystemeigene) LTIException, beim einem 404-Fehler eine Spezialisierung davon. Eine der internen "Wrapper-Methoden" zu diesem Zweck sieht beispielsweise wie folgt aus:

Der Sync-Job wird im Normalfall jeden Fehler (insb. jede LTIException) einfach nur protokollieren und am Ende eine Warnmeldung ausgeben, dass der Sync nicht fehlerfrei war.

Im "404-Fall", also im Fall, dass die entfernte Ressource in Moodle endgültig gelöscht wurde, sollte eine *Bereinigung* der Übungssystem-Datenbasis vorgenommen werden, d.h. alle im Übungssystem gespeicherten Resource-Links auf nicht mehr existierende Ressourcen sowie die davon abhängigen Daten wie <code>User</code>-Objekte zu den Resource-Links und Tripel zu den <code>User</code>-Objekten sollten gelöscht werden. Das bewirkt dann auch, dass nachfolgende Sync-Jobs nicht immer wieder denselben aussichtslosen Abgleich versuchen und dieselben Warnungen ausgeben.

Im Normalfall wird der Sync-Job aber sicherheitshalber nicht eigenmächtig über eine solche Löschung entscheiden (eventuell könnte der 404-Fehler ja doch ein Fehlalarm gewesen sein oder die Löschung ein Versehen gewesen sein und die fehlende Ressource aus einem Backup wiederhergestellt werden). Vielmehr wird nach Auftreten von 404-Warnungen in einem Job der entsprechende Whitelist-Eintrag (Kurs-Context-Paar) als "dirty" markiert, vgl. das "dirty"-Flag von Kurs__lti_optcontext in oben abgebildetem Klassendiagramm. In der Whitelist-Ansicht im Betreuer-UI werden solche Einträge dann entsprechend markiert.

Die folgende Abbildung zeigt das Betreuer-UI:

externes System ("Tool Consumer")	LTI-Kurs	Studentenrechte vererben ?	Betreuerrechte vererben ?	Outcomes ?	Trennen
Moodle22Test EVAL - EVAL	Immos Testkurs				8
Anzeige des Dirty-Flags in der Whitelist im Betreuer-Ul					

07.03.25 Seite 41 / 57

Der (\rightarrow) -Button in der Outcomes-Spalte ist nur bei aktivierter Outcomes-Nutzung vorhanden und dient zum manuellen Starten eines Sofort-Syncs zu genau diesem Whitelist-Eintrag. Der rote (!) -Button daneben wird nur angezeigt, wenn das Dirty-Flag des Eintrags gesetzt ist und startet einen *Sync-and-Clean-Job*.

Dieser *Sync-and-Clean-Job* ist ein vollständiger Sync-Job, der sich vom einfachen Sync nur dadurch unterscheidet, dass er bei 404-Fehlern die oben beschriebenen Löschaktionen zur Bereinigung durchführt, also "tote" Resource-Links im Übungssystem löscht.

UPDATE: Da, wie oben bereits eingeworfen, Moodle ab Version 3 auch bei fehlenden Ressourcen mit Code 200 antwortet, musste der Sync-Job so angepasst werden, dass er bei jeder LtiException das Bereinigungen unterstützt (nicht mehr nur bei 404-Codes).

Sync-and-Clean-Jobs werden – im Gegensatz zu Sync-Jobs – nur manuell und nur für einzelne Whitelist-Einträge angestoßen und nie als Cron-Jobs ausgeführt. Auf diese Weise ist sichergestellt, dass Löschungen aus der Übungssystem-Datenbasis von einem Menschen bestätigt werden müssen.

07.03.25 Seite 42 / 57

Embedded Views

Bei Einbindung einer »Externes Tool«-Ressource in Moodle kann der Benutzer unter »Startcontainer« die Art festlegen, wie die Toolseite angezeigt werden soll, insbesondere ob das Tool in einem eigenen Fenster oder in die Moodle-Seite eingebettet (als Inlineframe (iframe)

https://de.wikipedia.org/wiki/Inlineframe) geladen werden soll. Diese Entscheidung wird auch dem Tool als Parameter im Launch-Request mitgeteilt, so dass dieses seine Darstellung daran anpassen kann.

Der erste Abschnitt dieses Kapitels beschreibt die Implementierung eines (I)Frame-Modus im Online-Übungssystem. Der zweite Abschnitt geht auf per LTI übermittelte Return-URLs zur Rückkehr vom Tool zum Consumer ein und deren Einbindung im Übungssystem (im Frame-Modus). Die darauf folgenden Abschnitte gehen kurz auf die damit realisierten "Embedded Views" für Studentensicht und Betreuersicht auf Übungssystem-Aufgaben in per LTI-Request geöffneten Frames ein.

(I)Frame-Modus

Normalerweise sind die Webseiten (einschließlich der Aufgabenseiten) des Online-Übungssystems "vollständige" Webseiten, einschließlich verschiedener Navigationen wie Menüs und Brotkrümelpfad und einschließlich z.B. eines Kopfbereichs mit FernUni- und Übungssystem-Logos. Soll eine Übungssystem-Aufgabe dagegen in einem Frame innerhalb einer Moodle-Seite (die bereits selbst Kopfbereich, Navigation etc. umfasst) eingebettet werden, so sollten derartige Elemente nicht Teil des Frame-Inhalts sein. Vielmehr soll sich der Inhalt des Frames weitestgehend auf die Aufgabe selbst beschränken – wenn eine Aufgabe einzeln als Tool angeboten wird. Wird dagegen ein ganzes Aufgabenheft als Tool in einem IFrame angezeigt, muss der Frame-Inhalt auch eine Navigation zwischen den einzelnen Aufgaben des Hefts ermöglichen.

StudentenHeftStartSeite & StudentenHeftKorrekturSeite

Beim Start eines "Heft-Tools" (als Student) soll außerdem nicht sofort irgendeine Aufgabe angezeigt werden, sondern eine Übersichtsseite zum Heft, die insbesondere eine Aufgabenübersicht enthält, aber auch z.B. die Heft-Termine anzeigt und weitere Bearbeitungshinweise enthalten kann.

Das Online-Übungssystem bietet im Stand-Alone-Betrieb bereits eine Aufgabenübersichtsseite (StudentenStartSeite), die jedoch eine Übersicht über den gesamten Kurs (alle seine Aufgabenhefte, Bearbeitungstermine pro Heft, Aufgabenübersicht mit Anzeige des Bearbeitungsstatus etc.) bietet. Für den neuen LTI-Aufgabenheft-Tool-Modus wurde nun eine Variante davon eingeführt, die sich auf ein einzelnes Aufgabenheft beschränkt (StudentenHeftStartSeite).

Analog dazu wurde zur StudentenKorrekturSeite, die eine Übersicht über die Korrekturen aller Einsendungen (Anzeige des Korrekturstatus, Links zu freigegebenen Korrekturen) und aller verfügbarer Musterlösungen zum gesamten Kurs bietet, ebenfalls eine Aufgabenheft-spezifische Variante StudentenHeftKorrekturSeite eingeführt, die im LTI-IFrame angezeigt werden kann. Sie ist z.B. von der StudentenHeftStartSeite aus per Tab erreichbar, sobald Korrekturen oder Musterlösungen vorliegen).

07.03.25 Seite 43 / 57

Erweiterung des Embedding-Mechanismus des Online-Übungssystems

Das Online-Übungssystem wurde in den Neunzigerjahren so designed, dass vollständge Webseiten als HTML-Templates entworfen wurden, wobei es sich um "fast fertige" HTML-Dateien handelt, in denen jedoch bestimmte Variablen wie z.B. <code>\$KursNr</code> oder <code>\$Aufgabenname</code> vorkommen dürfen, die vor der Seitenauslieferung durch konkrete Werte aus dem Kontext ersetzt werden. Auch Aufgabenseiten sind (von den Aufgabenautoren direkt erstellte oder über Online-Aufgabeneditoren automatisch generierte) vollständige HTML-Templates.

Aus Abwärtskompatibilitätsgründen bestehen Aufgabenseiten, Quittungsvorlagen, Musterlösungsseiten etc. auch heute noch aus HTML-Templates, allerdings wurde der Template-Mechanismus stark erweitert, insbesondere um einen so genannten Embedding-Mechanismus. Dieser sieht vor, dass in einer Seitenressource (wie einem Aufgabentext) der eigentliche Content mit zwei Marken (\$EMBED und \$/EMBED) markiert wird, woraufhin das Online-Übungssystem versucht, diesen Inhalt in eine nach dem Kontext ausgewählte Seitenvorlage (sog. Page-Layout-Template) eingebettet wird. Dieser Mechanismus ermöglicht ein einheitliches Webdesign, da Seitenkopf, Navigationsmenüs etc. nicht redundant in den einzelnen Seitenressourcen vorliegen müssen.

Dieser Embedding-Mechanismus wurde nun wie folgt erweitert:

- 1. Erstellung spezieller Page-Layout-Templates für Frame-Inhalte, die auf Navigationsmenüs in einer Seitenspalte und den vollständigen Seitenkopf mit FernUni- und Übungssystem-Logo sowie Brotkrümelpfad etc. verzichten und im Wesentlichen den einzubettenden Inhalt darstellen ggf. mit Tabs zur Navigation zwischen verschiedenen Ressourcen zur selben Aufgabe (Aufgabentext, Korrektur, Musterlösung) und für LTI-Sessions mit X-Icon oben rechts, das zum LTI-Return-URL verlinkt.
- 2. Erstellung weiterer spezieller Page-Layout-Templates für Frame-Inhalte bei ganzen Aufgabenheften, die sich von den erstgenannten durch zusätzliche Heft-lokale Navigationsmöglichkeiten unterscheiden. So gibt es z.B. ein PageLayoutStudentAufgabeIFrameHeft, das für die Einbettung von Aufgabenseiten vorgesehen ist, einen Back-Link zur StudentenHeftStartSeite (also der Aufgabenübersicht zum Aufgabenheft) enthält sowie eine Navigationsleiste zum direkten Anspringen aller Aufgaben desselben Hefts, einschließlich Anzeige, welche der Aufgaben bereits bearbeitet wurden.
- 3. In der vom LTI-Launch-Request geöffneten Session wird ein Frame-Flag gesetzt (sofern der Tool-Consumer im Request ein "document target" in Moodle als »Startcontainer« übergibt, zu dem das Übungssystem dem IFrame-Modus aktiviert, Details folgen später im Abschnitt Return-Links derzeit nur im (I)Frame-Modus).
- 4. Falls sich der LTI-Launch-Request auf ein Aufgabenheft bezieht (im Launch-URL also keine Aufgabennummer angegeben wurde), wird ein weiteres Flag in der Session gesetzt, das die Frame-Interne Heft-Navigation aktiviert.
- 5. Die kontextabhängige automatische Wahl eines Page-Layout-Templates beim Embedding erkennt nun diese beiden Flags (IFrame ja/nein?, Heft-Navigation in IFrame ja/nein?) und wählt ein dazu passendes Page-Layout-Template aus.

07.03.25 Seite 44 / 57

Einschränkungen

Damit liegt auch folgende Einschränkung auf der Hand: Der spezielle (I)Frame-Modus steht nur für Seiten zur Verfügung, die den Embedding-Mechanismus nutzen. Bei Aufgabenseiten sollte das mittlerweile Standard sein. Alte Aufgabeseiten ohne Embedding enthalten aber meist ohnehin keinen nennenswerten Overhead, sondern i.d.R. nur die Aufgabenstellung und vielleicht noch einen Zurück-Link, und könnten notfalls ebenfalls in einem Frame dargestellt werden.

Es gibt einige Seiten im Online-Übungssystem, die den Embedding-Mechanismus *nicht* nutzen, denn der wurde vornehmlich für von Endnutzern (Kursbetreuern und Aufgabenautoren) erstellte Ressourcen vorgesehen. Manche Seiten, die in der Regel von den Endnutzern nicht selbst erstellt werden (wie Kursstartseite, Aufgaben- und Ergebnisübersichten für Studenten etc.) sind dagegen bereits statisch (per XSLT) in ein (XSL-)Seitentemplate eingebettet worden 12. Für solche Seiten steht der (I)Frame-Modus natürlich nicht zur Verfügung. Zumindest im LTI-Kontext ist letzteres aber keine relevante Einschränkung, da beim Einsatz als LTI-Tool nur Aufgabenressourcen im Frame anzuzeigen sind, solche Einstiegs- und Übersichtsseiten jedoch zur Übungssystem-lokalen Navigation gehören, die gar nicht in einem LTI-Tool angezeigt werden sollen. Auch für die neu eingeführte Einbindung ganzer Aufgabenhefte als LTI-.Tool werden diese kursübergreifenden Seiten nicht benötigt, vielmehr wurden heftspezifische Varianten davon eingeführt (StudentenHeftStartSeite & StudentenHeftKorrekturSeite), die ebenfalls Embedding unterstützen (und für die eigene Page-Layout-Templates für den IFrame-Einsatz erstellt wurden).

Sollte ein Benutzer innerhalb einer Session mit aktivem (I)Frame-Modus es dennoch schaffen, eine solche Übersichtsseite aufzurufen (die nicht lokal durch eine Variante mit Embedding-Support ersetzt wurde), so wird der (I)Frame-Modus beendet (das entsprechende Session-Flag entfernt). Denn in dem Fall wird von folgendem Szenario ausgegangen: Der User hat z.B. von Moodle aus eine Aufgabe (als IFrame) geöffnet. Später (aber in derselben Session) lädt er die Kursstartseite oder Aufgabenübersicht explizit in einem anderen Browserfenster. Dass diese im "Voll-Layout" dargestellt wird, ist dabei durchaus erwünscht. Falls er nun von hier aus wieder eine Aufgabe, Musterlösung, Korrektur o.ä. aufrufen sollte, soll auch diese wieder als vollständige Seite (also nicht im Frame-Modus) geladen werden. Ein ggf. noch gesetzes (I)Frame-Modus-Flag in der Session wird also wieder zurückgesetzt. Für spätere LTI-Toolaufrufe in Frames ist das unschädlich, denn ein neuer Launch-Request setzt auch das Frame-Flag wieder neu. Ein Konflikt ist nur denkbar, wenn ein User erst in einem Frame eine Aufgabe öffnet, dann in einem separaten Fenster den Frame-Modus beendet, dann im noch offenen Moodle-Frame z.B. seine Eingaben einsendet: Die Quittung im Frame würde dann als vollständige Seite gerendert. Das ist aber weder wahrscheinlich noch tragisch, denn der Content ist ja in jedem Fall vorhanden.

Query-Parameter

Bei LTI wird der (I)Frame-Modus, wie gesagt, direkt beim Launch Request gesetzt und in der Session vermerkt. Der Vollständigkeit halber sei an dieser Stelle aber auch vermerkt, dass der (I)Frame-Modus sich ebenfalls bei gewöhnlichen GET-Requests einschalten lässt, indem diesen der Query-Parameter iframe angehängt wird. Das könnte z.B. benutzt werden, falls von Moodle aus von LTI unabhängige Direktlinks zu (Frame-kompatiblen, d.h. Embedding nutzenden) Übungssystem-Seiten gesetzt werden sollen, die einen Frame als Target haben. (Wird ein per Get-Request eine StudentenHeftStartSeite oder StudentenHeftKorrekturSeite mit diesem Query-Parameter ?iframe=on) aufgerufen, so wird auch neben dem IFrame-Flag wieder das IFrame-Heft-Navigations-Flag gesetzt.

Auch für solche Links gelten dieselben Einschränkungen wie oben beschrieben.

07.03.25 Seite 45 / 57

Im Übrigen lässt sich der (I)Frame-Modus auch per Query-Parameter [iframe=off] wieder abschalten.

LTI-Return-URL

Der LTI-Standard https://www.imsglobal.org/specs/ltiv1p1/implementation-guide sieht vor, dass dem Tool ein Return-URL mitgeteilt wird, zu welchem er dem Nutzer einen Link anbieten sollte. Dieser führt wieder zur aufrufenden Umgebung (Moodle) zurück. Wenn neben dem (I)Frame die einbindende (Moodle-)Seite noch genügend Navigationsmöglichkeiten anzeigt, ist das vielleicht nicht ganz so wichtig, aber insbesondere auf Smartphones wird das Tool (die Übungssystem-Aufgabe) mitunter als volle Seite ohne umgebende Moodle-Elemente gestartet. Dieser Return-Link wird daher im IFrame-Page-Layout (siehe Embedding-Mechanismus) in Form eines Schließen-Symbols (X) in der oberen rechten Ecke eingeblendet.

Zu diesem Zweck wird der Return-URL mit in der vom Launch-Request geöffneten Session abgelegt und steht dort für das Rendering von Antwortseiten auch noch bei Folgerequests zur Verfügung. Das Page-Layout-Template prüft per Fallunterscheidung, ob ein solcher Return-URL in der Session liegt (was bei LTI-Sessions immer der Fall sein sollte, aber der Frame-Mechanismus könnte ja auch einmal unabhängig von LTI genutzt werden) und blendet, sofern vorhanden, das entsprechende Schließen-Symbol als Hyperlink auf diesen Return-URL ein.

Return-Links derzeit nur im (I)Frame-Modus

Das Einbinden des Return-Links erfolgt derzeit nur in den Page-Layout-Templates für die (I)Frame-Darstellung, in den normalen Vollbild-Templates wurde darauf verzichtet. Der Grund dafür ergibt sich u.a. aus folgenden Beobachtungen:

- Der Tool-Consumer sollte laut LTI-1.1 https://www.imsglobal.org/specs/ltiv1p1/implementation-guide im Launch-Parameter launch_presentation_document_target einen der drei Werte window, frame oder iframe übertragen.
- Das Online-Übungssystem aktiviert den (I)Frame-Modus, für die Werte frame und iframe, nicht jedoch für window.
- Moodle bietet für »Externes Tool« unter Option »Startcontainer« die folgenden
 Auswahlmöglichkeiten: »Eingebettet«, »Eingebettet ohne Blöcke«, »Vorhandenes Fenster«
 sowie »Neues Fenster«. Die ersten beiden Einstellungen betten einen IFrame ein,
 »Vorhandenes Fenster« lädt das Tool in Vollbild im Browserfenster und verlässt damit effektiv
 die Moodle-Umgebung (hier ist ein Return-Link also besonders wichtig!), und »Neues Fenster«
 öffnet ein Popup-Fenster, während im Original-Fenster die Moodle-Umgebung geöffnet bleibt
 (hier ergibt ein Return-Link nicht viel Sinn).
- Zumindest für Moodle ließen sich nun experimentell folgende Werte für obigen Parameter in Abhängigkeit dieser »Startcontainer«-Einstellung ermitteln:
 - »Eingebettet*« → [iframe]
 - »Neues Fenster« → window
 - »Vorhandenes Fenster« → frame
- Übrigens zeigte Moodle auch bei »Eingebettet«-Einstellung auf Smartphones das Verhalten, als sei »Vorhandenes Fenster« eingestellt, was auch eine sinnvolle Maßnahme im Rahmen eines Responsive Designs ist: Ein IFrame, der nur einen Teil des ohnehin schon kleinen Handydisplays zur Verfügung hätte, wäre einfach nicht gut bedienbar.

Zumindest für Moodle als Tool-Consumer gilt also demnach, dass die Vollbild-Page-Layout-Templates

07.03.25 Seite 46 / 57

nur zum Einsatz kommen, wenn das Tool als Popup-Fenster geöffnet wird, in dem ein Return-Link nicht benötigt wird, da die Moodle-Umgebung in einem anderen Fenster noch offen ist.

Besonders wichtig ist der Return-Link in Situationen, wo eigentlich kein IFrame geöffnet wird, sondern das Tool die Moodle-Umgebung ersetzt (»Vorhandenes Fenster«). In diesen Fällen stellt aber Moodle bereits sicher, dass der Frame-Modus genutzt wird, für welchen das Übungssystem wiederum den Return-Link einbindet.

Lediglich falls einmal andere Tool-Consumer-Systeme als Moodle eingesetzt werden sollten und in diesen das Tool auch absichtlich weder als Frame noch als neues Fenster, sondern in Vollbild im vorhandenen Fenster geöffnet werden soll, ist zu überprüfen, ob das Tool genau wie Moodle den frame-Parameter übermittelt oder etwa doch window. In letzterem Fall sollte auch im Vollbild-Page-Layout-Template ein Return-Link nachgerüstet werden – was übrigens nicht unbedingt Übungssystem global geschehen muss, sondern auch Kurs-lokal durch Hochladen kursspezifischer Page-Layout-Templates möglich ist.

Studentensicht

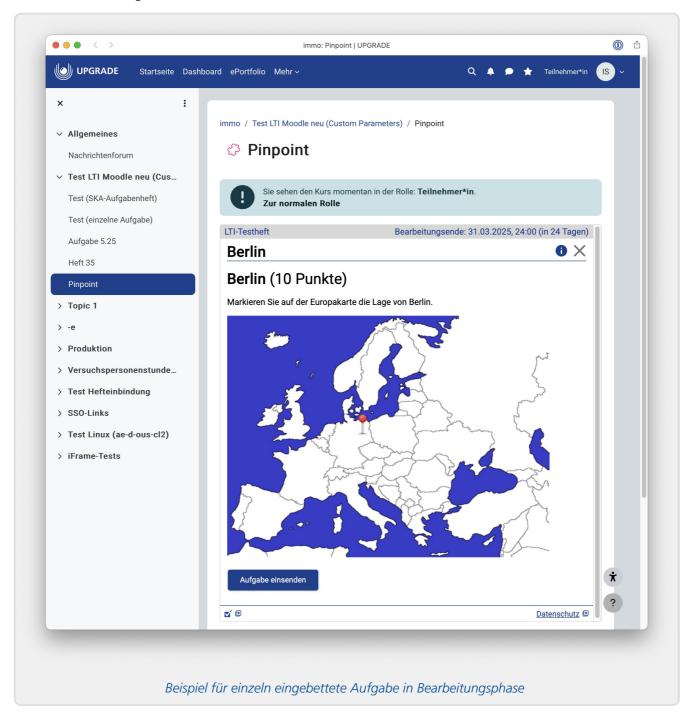
Dieser Abschnitt soll kurz darauf eingehen, wie die Ansicht eines LTI-Tools für einen Teilnehmer (Studenten) aussieht. Dabei gibt es leichte Unterschiede je nachdem, ob eine einzelne Aufgabe oder ein ganzes Aufgabenheft als Tool eingebunden wird.

Einzelaufgabe

In der *Bearbeitungsphase* (ab Bearbeitungsbeginn bis zum Bearbeitungsende¹³) enthält der Tool-Frame, wie er in Moodle eingebettet wird, lediglich...

- eine Überschrift,
- rechts oben ein Schließen-Icon, das auf den Return-URL verlinkt, also das Tool schließt und zu einer von Moodle bestimmten Seite zurückkehrt,
- daneben ein Info-Icon, über welches die Bearbeitungstermine des zugehörigen Aufgabenhefts (und somit der Aufgabe) eingeblendet werden können,
- und darunter das Aufgabenformular als Haupt-Inhalt des Frames.

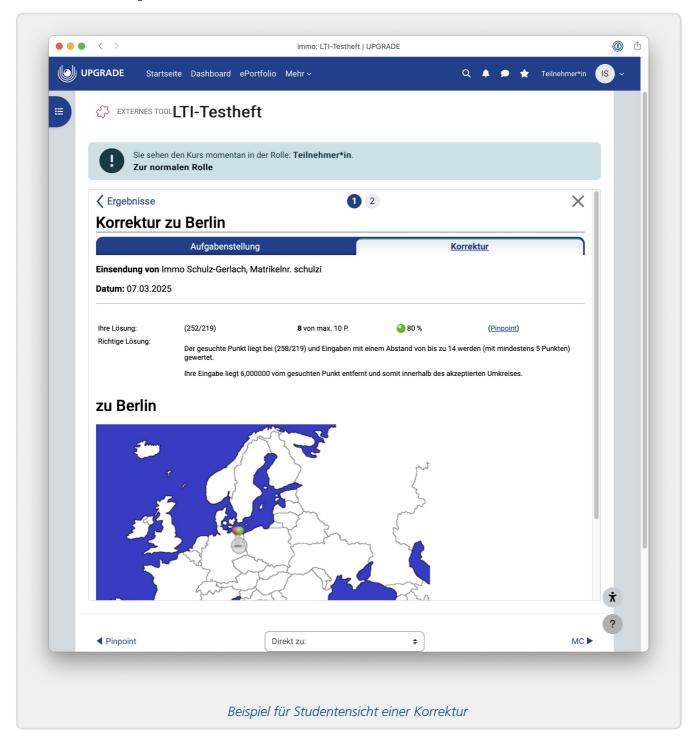
07.03.25 Seite 47 / 57



In *späteren Phasen* werden jedoch noch weitere Ressourcen verfügbar: Die Musterlösung (optional, typischerweise nach Einsendeschluss verfügbar) und die Korrektur (Autokorrekturen i.d.R. kurz nach Einsendeschluss, manuelle Korrekturen erst später). Diese sollen natürlich auch im Moodle-Kurs abrufbar sein und kein separates Aufrufen des Übungssystems in einem eigenen Browserfenster erfordern. Daher enthält der Frame ab Verfügbarkeit von Musterlösung oder Korrektur eine entsprechende *lokale* Navigationsmöglichkeit zwischen diesen bis zu drei Seiten, wofür hier "Tabs" als Darstellungsform gewählt wurden.

Der folgende Screenshot zeigt z.B. den Abruf einer Autokorrektur zu einer abgeschlossenen Aufgabe, zu der keine separate Musterlösungsseite hinterlegt wurde, da die Lösung hinreichend aus der Korrektur hervorgeht:

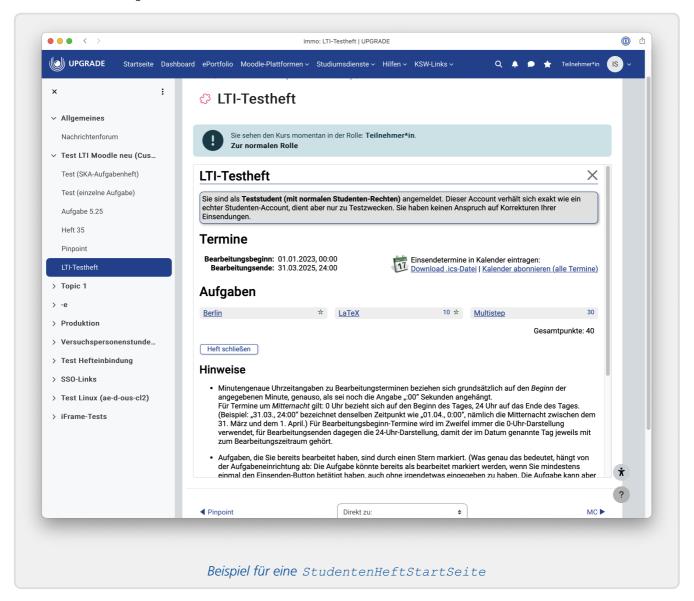
07.03.25 Seite 48 / 57



Aufgabenheft

Direkt beim Tool-Start enthält der Frame in diesem Fall die StudentenHeftStartSeite, also einen Übersichtsseite zum Aufgabenheft, von der sich insbesondere alle Aufgaben öffnen lassen.

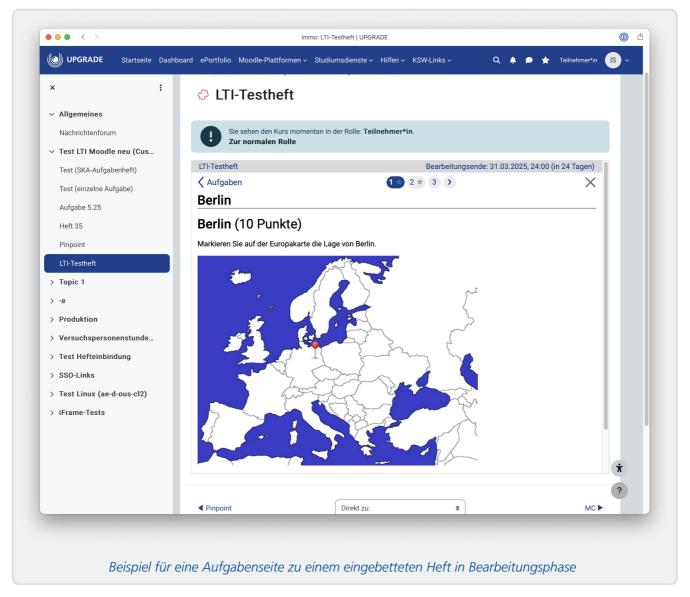
07.03.25 Seite 49 / 57



Bei Auswahl einer Aufgabe aus dieser Übersicht, wird die Aufgabenseite geöffnet. Diese sieht ähnlich aus wie bei Einzel-Einbettung der Aufgabe, nur mit zusätzlichen Navigationsmöglichkeiten:

- Oben links findet sich ein Zurück-Link, der wieder zur Übersichtsseite (StudentenHeftStartSeite führt),
- oben mittig findet sich eine Navigationsleiste, mit der direkt zu allen Aufgaben des Hefts gesprungen werden kann (wobei aus Platzgründen die Aufgaben hier nur durch ihre Nummern repräsentiert werden, der Aufgabenname wird aber als Tooltip angezeigt, wenn der Mauscursor über dem Link "hovert"),
- oben rechts findet sich wiederum der Close-Button, mit dem das LTI-Tool geschlossen werden kann.
- Auf ein Info-Icon zur Einblendung der Termine wurde hier dagegen verzichtet, da sich diese Termine in der Heftübersicht (StudentenHeftStartSeite) finden.

07.03.25 Seite 50 / 57



Auch hier gilt, dass *in späteren Phasen*, also sobald Korrekturen oder Musterlösungen verfügbar sind, entsprechende Tabs eingeblendet werden:

- Zwischen StudentenHeftStartSeite (Aufgabenübersicht) und StudentenKorrekturSeite (Übersicht über alle Korrekturen und Musterlösungen) kann dann über zwei Tabs gewechselt werden,
- und in allen Aufgaben-, Musterlösungs- und Korrekturseiten finden sich dann ebenfalls Tabs, mit denen zwischen der Aufgabenstellung, Korrektur und ggf. Musterlösung zur selben Aufgabe schnell gewechselt werden kann.

Betreuersicht

In Moodle können nicht nur Studenten, sondern auch Betreuer ein Tool aufrufen. Sie bekommen dann entsprechende Konfigurationsmöglichkeiten angezeigt, wieder abhängig davon, ob eine einzelne Aufgabe oder ein ganzes Aufgabenheft als Tool eingebunden wurde.

(Die LTI-Aufgabeneinbindung in Moodle aus Betreuersicht wird in einem separaten Handbuch https://online-uebungssystem.fernuni-hagen.de/download/LTI_Moodle/Uebungssystem_LTI_Moodle.html beschrieben.)

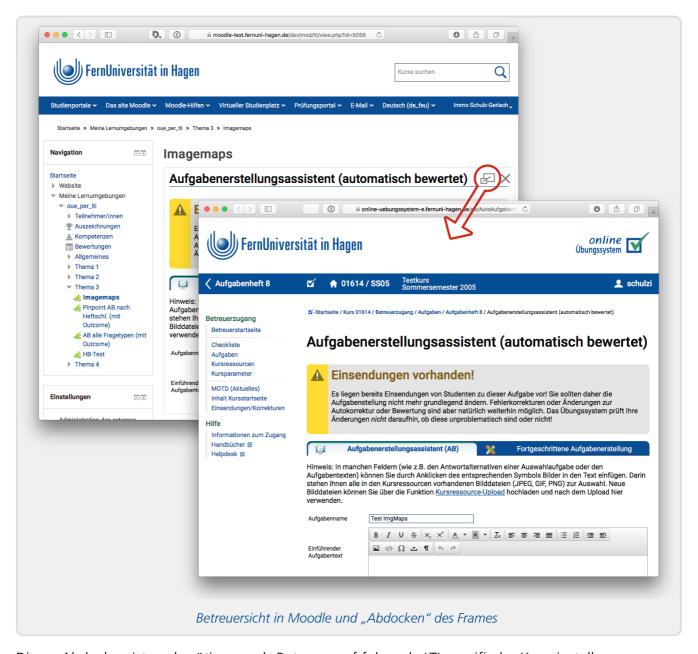
07.03.25 Seite 51 / 57

Einzelaufgabe

In diesem Fall wird dem Betreuer der Aufgabeneditor des Übungssystems im Frame angezeigt, so dass dieser sofort und ohne Moodle zu verlassen Änderungen an der Aufgabe vornehmen kann.

Darüber hinaus gibt es im Übungssystem aber noch deutlich mehr Stellschrauben, auf die der Betreuer Zugriff haben muss, z.B. zentrale Einstellungen ("Kursparameter"), Bearbeitungstermine der Aufgabenhefte etc. Diese stehen nicht im Tool-Frame zur Verfügung! Vielmehr soll dem Betreuer die Möglichkeit des Vollzugriffs auf das gesamte Übungssystem-Betreuer-UI seiner Kursumgebung geboten, dabei aber dennoch vom Single-Sign-On per LTI-Launch Gebrauch gemacht werden.

Dazu wurde folgende Lösung realisiert: Der Frame eines Betreuers enthält neben dem Aufgabeneditor und dem Schließen-Link mit Return-URL (s.o.) oben rechts noch ein weiteres Symbol zum "Abdocken" des Frames. Damit wird der Frame in der Moodle-Seite geschlossen (per Return-URL) und gleichzeitig ein neues Browserfenster geöffnet, das dieselbe Seite (den Aufgabeneditor) im vollständigen Übungssystem-Seitenlayout darstellt – und somit auch alle Navigationsmöglichkeiten bietet. Dabei wird die per LTI geöffnete Session mit Kurs-Autorisierung beibehalten, so dass der Betreuer sich nicht nativ im Übungssystem neu einloggen muss.



Dieses Abdocken ist auch nötig, um als Betreuer auf folgende LTI-spezifische Kurseinstellungen

07.03.25 Seite 52 / 57

Technische Doku: LTI-Integration Embedded Views

zugreifen zu können:

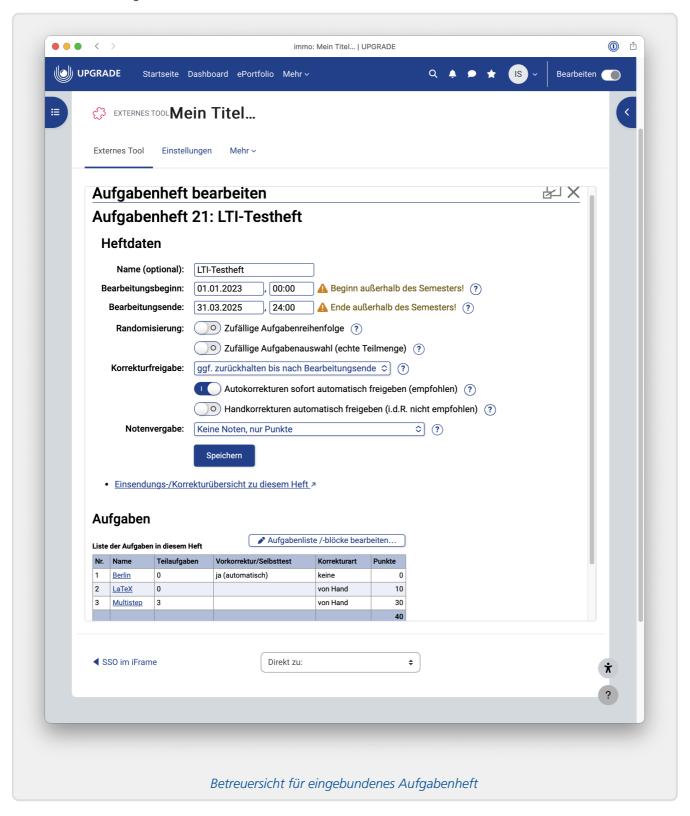
• Verwaltung der Whitelist (Menüpunkt »LTI-Verknüpfungen« auf der Betreuerstartseite), einschließlich Ein-/Ausschalten des Outcomes-Services und manuelle Sync-Jobs.

• In den Kursparametern lässt sich der LTI-Only-Modus ein-/ausschalten.

Aufgabenheft

Startet ein Betreuer ein als Tool eingebundenes Aufgabenheft, so wird ihm direkt die Aufgabenheftverwaltung angezeigt. Die Ansicht entspricht weitgehend der "gewöhnlichen" Ansicht des Online-Übungssystems, ist jedoch der Übersichtlichkeit halber in diesem Kontext auf das eingebettete Aufgabenheft beschränkt (d.h. es fehlen Navigationsmöglichkeiten zum Wechseln zu anderen Aufgabenheften).

07.03.25 Seite 53 / 57



Von hier aus kann der Betreuer z.B. die Bearbeitungstermine und andere Aufgabenhefteinstellungen bearbeite, die einzelnen Aufgaben zur Bearbeitung öffnen, weitere Aufgaben hinzufügen etc.

Auch hier wird – wie bei einer Einzelaufgabe als Tool – die Möglichkeit zum "Abdocken" des Betreuerinterfaces geboten, über das auch andere Betreuertätigkeiten wie z.B. Bearbeiten der Kursparameter oder LTI-Einstellungen ermöglicht werden.

07.03.25 Seite 54 / 57

Wartungshinweise

DNS-Umbenennung eines Tool-Consumers

Im September 2017 wurden an den FernUni-Moodle-Installationen Sub-Domain-Umbenennungen durchgeführt (z.B. von moodle2wrm.fernuni-hagen.de) zu moodle-wrm.fernuni-hagen.de). Diese hatten bei bestehenden LTI-Connections prompt zu zwei Sorten von Fehlern geführt, die administrativ behandelt werden mussten. Auch andere Situationen könnten ggf. ähnliche Symptome auslösen.

- Ein Tool-Consumer kann eine selbst generierte GUID mitsenden. Moodle generiert hier nicht etwa eine zufällige Zeichenfolge, sondern verwendet ganz einfach seinen Domainnamen als GUID. (Das muss nicht bei jedem anderen Consumer-System so sein.) Die Folge: Durch die Domain-Umbenennung ändert sich auch die GUID. Für bei GUID-Änderung zu ergreifende Maßnahmen siehe folgenden Abschnitt.
 - Wichtig: Es ist daher absolut davon abzuraten, eine Moodle-Installation unter verschiedenen Alias-DNS-Namen anzubieten, denn dann würde je nach vom User verwendeten DNS-Namen das Moodle als Tool-Consumer unterschiedliche GUIDs senden (obwohl es sich um dasselbe System handelt)! Bei den FernUni-Moodles wurde für die Übergangszeit auf dem alten Domainnamen einfach ein Redirect auf den neuen geschaltet, der das Problem vermeidet. Andernfalls müsste notfalls im Übungssystem die »Protected«-Option in den Tool-Consumer-Einstellungen abgeschaltet werden, welche bewirkt, dass alle Requests mit identischer GUID erwartet werden und Zugriffe mit von der gespeicherten abweichender GUID abgelehnt werden.
- 2. Ändert sich der Hostname des Tool Consumers, so ändert sich auch die WebService-URL für z.B. den Outcomes-Service der Moodle-Instanz. Da das Übungssystem zu bestehenden Einsendungen, deren Kurse den Outcomes-Service nutzen, die Service-URLs cacht, werden ohne Update erst einmal nach der Hostname-Änderung die Outcomes-Aufrufe (z.B. im Sync-Cronjob) fehlschlagen. Die gecachten URLs müssen daher aktualisiert werden.

Änderung der Tool-Consumer-GUID

Der LTI-Standard sieht vor, dass ein Tool-Consumer im Launch-Request eine GUID im Parameter tool_consumer_instance_guid übertragen sollte. Das wird *empfohlen*, aber nicht zwingend vorgeschrieben.

Die Tool-Provider-Library speichert beim ersten Launch-Request eines Tool-Consumers diese GUID des Consumers in der DB (Tabelle lti_consumer) ab – sofern der Parameter übertragen wird. Wenn in den Consumer-Einstellungen nun die Option »Protected« aktiviert wird, werden nachfolgende Launch-Requests dieses Consumers nur noch akzeptiert, wenn diese den GUID-Parameter ebenfalls mit übertragen, und zwar mit demselben, in der DB gespeicherten Wert.

Sollte ein Consumer seine GUID im Betrieb einmal ändern, werden bei aktivertem "Protected-Mode" dementsprechend alle Launches abgelehnt.

In diesem Fall bietet das Übungssystem in der Consumer-Administration https://online-uebungssystem-e.fernuni-hagen.de/Admin/LtiConsumers die Möglichkeit, die gespeicherte GUID zu löschen. Danach ist ein neuer Launch-Request irgendeines Tools von diesem Consumer auszuführen, damit die neue GUID gespeichert wird.

07.03.25 Seite 55 / 57

Anderung Outcomes-Service-URLs

Die LTI-Tool-Provider-Library speichert (cacht) in der Übungssystem-DB (Tabelle lti_context) zu jedem Resource-Link, für den min. einmal ein Launch-Request ausgeführt wurde, den Service-URL für den Outcomes-Service. Bei jedem Launch-Request wird dieser aktualisiert.

Ändert sich der Service-URL (z.B. durch DNS-Umbenennung), stimmen die gecachten Service-URLs nicht mehr. Bei Kursen mit Outcomes-Nutzung werden dann vorerst die Aufrufe des Outcomes-Services wegen des fehlerhaften URLs scheitern. Der Outcomes-Sync-Cronjob wird entsprechend viele Warnungen loggen und auch vorschlagen, dass im Falle, dass die Fehlermeldung auf permanente Fehler hindeutet oder die korrespondierende Ressource im Tool-Consumer gelöscht wurde, ein Sync-and-Clean-Job ausgeführt werden sollte. Das ist hier nicht zu empfehlen! Ein Clean-Job würde die Referenzen der einzelnen Studenten auf ihre Gradebooks ganz löschen, statt den zugehörigen URL zu aktualisieren!

Vielmehr sind die ResourceLinks zu aktualisieren. Das passiert, wie oben bereits gesagt, bei jedem Launch-Request. Das heißt: Für jeden betroffenen Resource-Link (z.B. eine »Externes Tool«-Ressource in Moodle) ist einmalig mit irgendeinem Account (egal ob Teilnehmer- oder Betreuerrolle) das Tool aufzurufen / zu starten. Ab sofort funktionieren dann wieder alle Outcomes-Aufrufe für alle vorhandenen Ergebnisse zu dieser Ressource/Aufgabe.

Für welche Resource-Links ein solcher "Update-Launch" ausgeführt werden muss, kann am einfachsten direkt der Liste von Warnungen im Log des Sync-Cronjobs entnommen werden. Zur Kontrolle wird abschließend ein neuer Sync-Lauf gestartet: Läuft dieser nun ohne Wartungen durch, wurden alle benötigten Updates vorgenommen, andernfalls nennt das Log die noch nicht funktionierenden Resource-Links.

Fußnoten

- 1. WebAssign ist ein CampusSource-Projekt https://www.campussource.de/software/webassign/, das Online-Übungssystem ist ein Fork des ZMIs von WebAssign mit diversen FernUni-spezifischen Schnittstellen zur FernUni-Systemlandschaft, durch konsequente Weiterentwicklung aber auch in vielen anderen Punkten inzwischen stark vom Original-WebAssign abweichend.
- 2. Tatsächlich stellt das Übungssystem als Tool-Provider geringfügig über den LTI-Standard hinausgehende Anforderungen, die von Moodle erfüllt werden, aber ggf. nicht von jedem LTI-Tool-Consumer. Insb. wird erwartet, dass der LDAP-Loginname des in Moodle eingeloggten Benutzers in einem Extension-Parameter des Launch-Requests mitgesendet wird. Das wird später noch genauer erläutert.
- 3. Eventuell könnte auch TSUGI die Voraussetzungen erfüllen, vielleicht sogar besser, das war aber auf den Webseiten nicht direkt zu erkennen. Dazu müsste man sich intensiver damit beschäftigen, detaillierter in TSUGI einsteigen, und die Bezeichnung als Framework für Multi-Tenant-Lernplattformen klingt so, als ob das entweder gar nicht so einfach geht oder das Framework zumindest sehr viel Ballast mitbringt, den ich nicht bräuchte. Die Tool-Provider-Library dagegen ist sehr gut überschaubar.
- 4. LTI 2.0 überarbeitete die Outcomes-Schnittstelle übrigens nochmals, nutzt z.B. JSON- statt XML-Nachrichten, aber durch die Wahl dieser LTI1-Library ist das Online-Übungssystem vorerst auch hier auf LTI 1.1 oder 1.0 mit Outcomes-Extension festgelegt. Das ist derzeit keinen Einschränkung, da LTI 1.1 alle von uns benötigten Features liefert, und technische Details wie die Erzeugung der XML-Nachrichten werden ohnehin von der Library komplett verkapselt.
- 5. Sonderfall: Sollte ein Veranstalter (z.B. ein Lehrgebiet) im Einzelfall eine eigene, lokale Moodle-

07.03.25 Seite 56 / 57

Installation betreiben, also nicht auf ein zentral vom ZMI gewartetes Moodle zurückgreifen, und soll das mit dem Übungssystem per LTI verbunden werden, so muss sich der Moodle-Administrator ans ZMI wenden, um für seine Moodle-Instanz neue Zugangsdaten (Key, Shared Secret) zu erfragen, mit denen er dann selbst eine entsprechende Tool-Konfiguration einrichten kann

- 6. Der Name "Versionsnummer" rührt daher, dass für jedes Semester eine neue semesterspezifische Kursversion angeboten wird, sich also die Inhalte pro Semester unterscheiden können
- 7. Das ist natürlich nur der URL für das Produktionssystem, bei Test- oder Staging-Installationen abweichend. Wichtig ist auch, dass nicht *irgendein* Hostname des Systems verwendet wird, sondern der *primäre*, hier also online-uebungssystem und nicht etwa online-pruefungssystem!
- 8. In den Moodle-Kursen der FernUni gibt es derzeit zwei verschiedene Betreuerrollen: Mit und ohne Bearbeitungsrechte. Allerdings sendet Moodle für beide Rollen beim Launch-Request denselben Parameter role=Instructor. Unter diesen Bedingungen kann auf Übungssystem-Seite naturgemäß auch keine Unterscheidung stattfinden, beide Rollen bekommen daher vollen Betreuerzugang im Online-Übungssystem. (Möglicherweise lässt sich die LTI-Rolle in Moodle konfigurieren, so dass hier ggf. zukünftig eine unterschiedliche Behandlung beider Rollen ermöglicht werden könnte.)
- 9. Wir setzen also voraus, dass Übungssystem-Aufgaben in Moodle-Kursumgebungen nur eingesetzt werden, wenn die studentischen Teilnehmer reguläre FernUni-Studenten mit einem zentralen FernUni-Account sind. Das ist aber ohnehin der Regelfall. (Sollten im Ausnahmefall in Moodle lokale Accounts angelegt und zu einem Moodle-Kurs mit Übungssystem-Aufgaben angemeldet werden, so müssen im Übungssystem-Kurs identische lokale Accounts manuell angelegt werden.)
- 10. Wie schon eingangs erwähnt, wird hier der Outcome-Service des LTI-Standards in Version 1.1 verwendet, in welchem die Nachrichten noch XML-kodiert sind. LTI 2 verwendet dagegen JSON-Nachrichten.
- 11. Die Art des Frames, ob also Teil eines HTML-Framesets oder Inlineframe/iframe, spielt dabei prinzipiell keine Rolle, daher sei i.F. nur kurz von Frames die Rede.
- 12. Einige dieser Seiten können im Bedarfsfall dennoch von Kursbetreuern ausgetauscht werden. Zu dem Zweck können die Betreuer ein Designpaket mit (XSLT-generierten) Vorlagen dieser Seiten in verschiedenen Ausprägungen herunterladen und ausgewählte Varianten davon in ihren Kurs hochladen, diese bei Bedarf sogar vorher noch manuell anpassen.
- 13. Prinzipiell bietet das Online-Übungssystem auch die Möglichkeit, dass ein Student seine Lösungen vorzeitig abgeben kann, so dass sie früher korrigiert werden können. In dem Fall würde seine individuelle Bearbeitungsphase schon vor Bearbeitungsende/Einsendeschluss enden. Im LTI-Kontext ist das derzeit aber nicht vorgesehen, der benötigte Heft-Schließen-Knopf ist nicht Teil einer einzelnen Aufgabenansicht, sondern nur der zentralen Aufgabenübersichtsseite des Übungssystems welche wiederum in LTI-Frames gar nicht angezeigt wird.

07.03.25 Seite 57 / 57