Moodle-Plugin Lernstrategien

Software Dokumentation

metromorph softworks GmbH

Inhalt

1. Überblick

1.1 Aktualität

Die verlinkten Seiten spiegeln den **aktuellen**[1] Entwicklungsstand von Moodle wider. Die Codebase des Plugins ist in Teilen älter.
Mit der Pluginversion 1.0.4 (2023120601) wurde ein großer Teil des Codes refactored und an die Anforderungen von Moodle 4.1 angepasst.

Das Plugin ist nur in der Version 4.1 von Moodle lauffähig.

Diese Dokumentation bezieht sich auf die Version 1.0.4 (2023120601) des Lernstrategien-Plugins.

1.2 Struktur der Inhalte

Das Plugin stellt einen Moodle-Kurs inlusive der Inhalte zur Verfügung.

Dabei übernimmt das Plugin vollständig die Ablaufsteuerung. Inhalte und Ablauf variieren auf Grundlage der gegebenen Anworten.

1.2.1 States

Eine zentrale Rollen spielen dabei die sogenannten States, die jeweils einen Schritt im Kursablauf darstellen. Ein State hat in der Regel eine Darstellung durch eine Kursseite, es gibt aber auch States, die nicht zur Darstellung gelangen und nur Steuerfunktionen haben.

1.3 Grundlegende Funktionsweise

Das Plugin ist vom Typ Activity Modules und verwendet zur Seitendarstelltung

- Mustache-Templates,
- Javascript in Form von AMD-Modulen und
- CSS.

Die Datei `view.php` stellt den Hauptprozess zur Verfügung. Unterroutinen sind in der Datei `/classes/lib.php` ausformuliert.

Die Statelogik (welcher State ist unter welchen Bedingungen der nächste) ist in `/classes/statemachine.php` implementiert.

Die Rendering-Routinen greifen auf die Mustache-Templates im Verzeichnis `/templates` zurück.

Die Nutzer-Antworten werden als JSON-String in der Datenbank (`splernstrategien_answers`) abgelegt.

1.4 Lizenz der Dokumentation

1. Stand Oktober 2023 ↩

2. Lizensierung des Plugins

Der Selbstlernkurs „Lernstrategien“ wird von ORCA.nrw folgendermaßen offen lizenziert und der Öffentlichkeit zur Verfügung gestellt:

2.1 Inhalte

Alle Inhalte wie Texte, Bilder, Frage-Antwort-Konstellationen und die didaktische Konzeption werden mit einer CC BY-SA 4.0 Lizenz zur Verfügung gestellt; ausgenommen sind verwendete Marken (Logos) und andere Elemente, welche entsprechend gekennzeichnet sind. Unter den Bedingungen der CC BY-SA 4.0 Lizenz – also insbesondere unter Beachtung der TULLU-Regel – ist eine Nachnutzung demnach ausdrücklich erlaubt und erwünscht. Anzugebender Urheber der Inhalte sind „Ferdinand Stebner & die AG Erziehungswissenschaft mit dem Schwerpunkt Pädagogische Diagnostik und Beratung der Universität Osnabrück“.

2.2 Programm

Zugang zu den Inhalten gewährt die Webseite ORCA.nrw. Für weitergehende Nachnutzungen steht „Lernstrategien“ als moodle-Plugin separat zur Verfügung (Dateiname: „mod_splernstrategien“). Das moodle-Plugin ist mit der GNU General Public License Version 3 lizensiert. Sie können es unter den Bedingungen der Lizenz – Version 3 oder (nach Ihrer Wahl) jeder neueren veröffentlichten Version – weiter verteilen und/oder modifizieren. Anzugebender Urheber des Plugins ist die „metromorph softworks GmbH im Auftrag von ORCA.nrw“. Mitanzugeben ist immer auch die Lizensierung der Inhalte durch die CC BY-SA 4.0 Lizenz.

3. Installation

3.1 Modul

Das Modul Lernstrategien liegt in der Regel als Zip-Datei vor.

In Ihrer Moodle-Instanz rufen Sie als Administrator die Seite

`admin/tool/installaddon/index.php`

auf und fügen dort die Zip-Datei ein.

Alternativ entpacken Sie den Inhalt der Zip-Datei im Verzeichnis `/moodle/mod/splernstrategien`.

Sie werden durch die Installation des Moduls geführt.

Das Modul kann dann als Aktivität in einem Kurs eingebunden werden.

4. Datenbank

4.1 Installation und Updates

Bei der Installation und Updates des Plugins werden die Datenbanktabellen gewartet. Dabei kommt die Upgrade-API von Moodle zum Einsatz.

4.2 Struktur und Inhalte der Tabellen

Voraussetzungen aus den Datenbank-Vorgaben von Moodle wurden in der ursprünglichen Version (2019) eingehalten.

Das Plugin verwendet folgende Tabellen:

- `mdl_splernstrategien`
- `mdl_splernstrategien_answers`
- `mdl_splernstrategien_modules`
- `mdl_splernstrategien_users`

Beachte: Das Tabellenpräfix (hier: `mdl`) ist von der Moodle-Instanz abhängig.

4.2.1 Tabelle `mdl_splernstrategien`

Diese Tabelle enthält Informationen zu den Einbindungen der Aktivität in Kursen.

4.2.2 Tabelle `mdl_splernstrategien_answers`

Enthält die Antwortdaten je Nutzer und Kurs.

Spalte	Typ	Verwendung
`id`	`bigint(10)` Auto-Inkrement	
`user_id`	`bigint(10) [0]`	User ID
`course`	`bigint(10) NULL [0]`	Kurs-ID (aus Moodle)
`moodle_id`	`bigint(10) NULL`	Instanz (Kontext, aus Moodle)
`data`	`longtext NULL`	Nutzer-Antworten als JSON-String

4.2.3 Tabelle `mdl_splernstrategien_modules`

State-Definitionen. Siehe in den Erläuterungen dazu.

Wenn sich bei einem Update in den States etwas ändert (in der Datei `/classes/states.php`), muss diese Tabelle während des Upgrades geleert werden. Dies wird durch einen Eintrag in der Datei `/db/upgrade.php` sichergestellt. Vergleiche hierzu die Kommentare in dieser Datei.

4.2.4 Tabelle `mdl_splernstrategien_users`

Informationen zum Bearbeitungsstand eines Nutzers je Kurs.

Spalte	Typ	Verwendung
`id`	`bigint(10) Auto-Inkrement`	
`course`	`bigint(10) NULL [0]`	Kurs-ID (aus Moodle)
`moodle_id`	`bigint(10) NULL [0]`	Instanz (Kontext, aus Moodle)
`timecreated`	`bigint(10) NULL [0]`	Anlagezeit
`user_id`	`bigint(10) NULL [0]`	Nutzer-ID (Moodle)
`laststate`	`bigint(10) NULL [0]`	Letzer abgeschlossener State
`nextstate`	`bigint(10) NULL [0]`	Nächster State
`epa_finished`	`smallint(4) NULL [0]`	Ist EP-A abgeschlossen?
`epb_finished`	`smallint(4) NULL [0]`	Ist EP-B abgeschlossen?
`wp_finished`	`smallint(4) NULL [0]`	IST WP abgeschlossen?
`delay`	`bigint(10) NULL [0]`	Verzögerung in Sekunden
`timestamp`	`bigint(10) NULL [0]`	Start des Delays

5. States

5.1 States

Die States werden durch Einträge in der Datenbank definiert.

Ein State definiert seinen Folgestate entweder direkt oder als Ergebnis auf eine Nutzereingabe.

States, die im Userinterface dargestellt werden, haben ein Template-File.

5.1.1 Definition von States in der Datenbank

Die Eigenschaften der States (Funtionsweise, Darstellung,...) werden in der Datenbank in der Tabelle `mdl_splernstrategien_modules` definiert.

Beim ersten Aufruf des Plugins, werden diese Daten aus der Datei `classes/states.php` ausgelesen und in die Datenbank übertragen.

Die Tabelle hat folgende Struktur:

Name	Typ und Defaultwert	Verwendung
`id`	`bigint(10)` Auto-Inkrement	eindeutige ID
`moodle_id`	`bigint(10) NULL [0]`	
`state`	`bigint(10) NULL [0]`	Nummer des States
`type`	`longtext NULL`	Art des States: `abfrage, modulauswahl checkingTraces, delay, abfrageVideo, check, trigger1, trigger2, trigger3`
`delay`	`bigint(10) NULL [0]`	Verzögerung in Sekunden
`strang`	`smallint(4) NULL [0]`	Strang in der Basiseinheit
`abschnitt`	`longtext NULL`	Abschnitt
`name`	`longtext NULL`	Name des States
`description`	`longtext NULL`	Beschreibung des States
`waysout`	`longtext NULL`	Mögliche nächste States (Komma-separiert)
`wayback`	`longtext NULL`	voriger State
`template`	`longtext NULL`	Templatefile
`end_strang`	`tinyint(2) NULL [0]`	Strangende
`questions`	`longtext NULL`	JSON mit Formular-Definitionen

5.2 Berechnung des nächsten States

5.2.1 Ohne Auswahl

Wenn in der State-Definition die Variable `waysout` nur einen Wert enthält, wird dieser State direkt aufgerufen.

5.2.2 Mit Auswahl

Enthält `waysout` mehrere Werte, kommt es auf den Inhalt der Variablen `questions` an. Dabei kann die Nutzereingabe in ein Textfeld ausgewertet werden, oder ein selektierter Radio-Button bestimmt den nächsten State.

Aus der Eingabe in ein Textfeld.

In der Definition des aktuellen States muss das Textfeld, auf das es ankommt, in `questions` beschrieben sein. Z.B.:

```
'questions'=>'{
  "data": {
      "0": {
          "type": "text",
          "name": "automarken_nach_land",
          "waysout": {
            "0": {
              "out": 2,
              "values": ["Audi", "Mercedes", "VW" ]
            },
            "1": {
              "out": 3,
              "values": ["Ford", "Chevrolet"]
            },
            "1": {
              "out": 4,
              "default": "true"
            }
          }
      }
  }
}'
```

Eine Eingabe von "Audi" leitet dann auf den State 2 weiter.

Aus der Auswahl eines Radio-Buttons

Die Radio-Gruppe muss im state beschrieben sein. Z.B. für eine Gruppe mit dem Namen "wohin" und den Optionen "weg", "nachhause" und "arbeit":

```
'questions'=>'{
  "data": {
     "0": {
         "type": "radio",
         "name": "wohin",
         "options": {
             "0": {
               "name": "weg",
               "out":"16"
             },
             "1": {
               "name": "nachhause",
               "out":"32"
             },
             "2": {
                "name": "arbeit",
                "out":"48"
             }
         }
     }
  }
}'
```

Je nach Wahl der Option wird man in den State 16, 32 oder 48 geleitet.

5.3 Templates

Die Inhalte der States werden durch Mustache-Templates repräsentiert.

Für jeden State, der eine Darstellung als Seite besitzt liegt in `templates` eine Datei mit dem Namen `template_<statenumber>.mustache`.

Im Rendering-Prozess wird das Template des jeweiligen States aufgerufen und die darzustellende Seite damit gebaut.

Wiederkehrende Elemente (Buttons, Ja-Nein-Abfragen) werden durch Partials dargestellt und in den State-Templates eingebunden.

Die Übergabe von variablen Inhalten (Nutzereingaben, Ergebnisse,...) erfolgt im PHP-Code mittels Templatevariablen.

Dabei wird das Moodle-eigene Template-Rendering verwendet.

6. Medien

6.1 Dateien

Die notwendigen lokalen Mediendateien liegen im Verzeichnis `material`.

```
material
└── motivationsregulation
    ├── audio     Podcasts
    ├── img       Bilddateien
    ├── pdf       PDF- Dokumente (Literaturlisten, Modelle)
    └── videos    Vorschaubilder und Untertitel zu den Videos
```

Videos werden von extern eingebunden.

Zum Ändern der Videoquellen müssen in allen Video-Tags die `src`-Attribute auf die neue Quelle geändert werden. Die zu ändernden Dateien findet man mit
`grep "<video" splernstrategien/templates/* -l` (24 Dateien)
In den Dateien sucht man die `<video>`-Tags und bearbeitet sie:

```
<video controls="true" controlsList="nodownload" style="width:100%; height:100%;"
    title="Strategie zur Motivationsregulation"
    poster="./material/motivationsregulation/videos/WP_Vignette1.jpg">
  <source
      src="https://cdn.educast.cloud/vod/orca/export/a6c433de-47b0-473d-9c6b-44be636837c8/3c180c96-9b3f-4815-a464-a40a181b6368/5596486.segment-0-1080p-sdr8-stdfps.mp4">
           ^^^^^^^^ diesen Eintrag bearbeiten
  <track
      src="./material/motivationsregulation/videos/untertitel/untertitel_lernstrategien_14_antworten_strang_3_a.vtt"
      kind="subtitles"
      srclang="de"
      label="Deutsch">
</video>
```

Wenn sich am Inhalt des Videos etwas ändert, sollte man in Erwägung ziehen `poster` und `track-src` anzupassen. Die zugehörigen Dateien ind im Plugin.

6.2 Wiedergabe

Die Medienwiedergabe von Videos und Audios erfolgt über die browsereigenen Mechanismen (HTML5-Tag).

7. CSS-Styling

Alle Inhalte des Plugins befinden sich innerhalb von

```
<div id="lernstrategien" class=lernstrategien">

</div>
```

insofern sind alle Styles mit der Klasse `lernstrategien` prefigiert.

Bis auf das spezielle Styling einzelner States sind alle plugin-spezifischen Styles in der Datei `/css/lernstrategien.css`.

8. Erzeugung von PDF-Dateien

In einigen States werden die Nutzereingabgen als PDF zum Download angeboten. Die Erzeugung der PDF-Dateien geschieht in `viewPDF<x>.php` unter Verwendung der TCPDF library, einer Bibliothek zum Umwandeln von HTML nach PDF, die von Moodle mitgeliefert[1] wird.

1. vergl. Moodle Quellcode ↩

9. Javascript

Der Code des Plugins enthält einige Javascript Module und verwendet wiederum einige Module aus dem Core von Moodle.

9.1 Aktualität

Der Javascript-Code ist in Form von AMD-Modulen realisiert. Insgesamt geht die Entwicklung von Moodle in Richtung ES-Module[1]. Dabei sollte in Zukunft weitgehend auf jQuery verzichtet werden.[2]

Dies erfordert in Hinblick auf Moodle 4 ein teilweises Refactoring des Javascript-Codes im Plugin.

- `amd/src/list_sort.js`
- `amd/src/dragdrop.js`

Diese Scripte brauchen jQuery, weil die zugrunde liegende Moodle-Bibliothek SortableList[3] auf jQuery aufbaut. Im Interface von SortableList kommen jQuery-Objekte vor (z.B. Events).
Erst wenn die moodle-eigenen Libs kein jQuery mehr benötigen, macht es Sinn, das auch hier zu entfernen.

9.2 Entwicklung

Die Entwicklung der JS-Module erfolgt unter Einsatz einer minimalen Grunt-Installation, die das Minifying erledigt.

Der Quellcode der Module liegt unter `amd/src`, der minifizierte Code unter `amd/build`

```
amd
├── build
│   ├── abfrage_video.min.js
│   ├── dragdrop.min.js
│   └── list_sort.min.js
└── src
    ├── abfrage_video.js
    ├── dragdrop.js
    └── list_sort.js
```

Das Minifying erfolgt dann mit dem Aufruf:

```
npx grunt amd
```

bzw. ohne Linting mit:

```
npx grunt uglify
```

---

1. vergl.: Modules | Moodle Developer Resources ↩
2. vergl.: jQuery | Moodle Developer Resources ↩
3. The Moodle UI Component library ↩

10. Mailversand

Das Plugin verwendet Notifications um den Nutzenden den Ablauf der Wartezeit vor den Triggereinheiten mitzuteilen. Notifications werden über die Weboberfläche und per E-Mail ausgeliefert.

10.1 Einstellungen

Folgende Einstellungen sind zu treffen:

- **Notifications aktivieren**:
  Unter `Administration -> General -> Messaging -> Notification Settings (/moodle/admin/message.php)` sollten Web und Email aktiviert sein. Und auch die Einstellungen für das Plugin sollten aktiviert sein. Diese geben den Default-Wert für die Nutzereinstellungen an. Wenn man die Sperre aktiviert, können die Nutzenden diese Einstellung in ihrem Konto nicht verändern.
- **Mailserver**:
  `Site-administration -> Server ->Email -> outgoing mail configuration (/moodle/admin/settings.php?section=outgoingmailconfig)`
  Der Mailserver, der die Mail entgegennimmt und weiterverteilt muss hier ggf. mit Port eingetragen sein (`mail.example.com:25`).
- **Cron muss laufen**:
  Der Cron[1] muss jede Minute einmal das Script in admin/cli/cron.php ausführen.
- **Die Taskzeiten berücksichtigen, oder einstellen**:
  Unter `Site-Administration -> Server -> Tasks -> Scheduled Tasks (/moodle/admin/tool/task/scheduledtasks.php)` nach `splernstrategien` suchen.
  Per Default ist Minute 2 jeder sechsten Stunde eingestellt (0:02, 6:02, 12:02,...). Das heißt, Mails werden per Default nur alle sechs Stunden verschickt.
  Zum Testen sollte man jede Minute einstellen (`*` in allen Feldern).
- Die Nutzenden müssen eine gültige Mailadresse haben.
- Das Ende des Delays muss erreicht sein.

10.2 Zur Implementierung

Grundlagen:

- https://docs.moodle.org/dev/Message_API
- https://moodledev.io/docs/4.1/apis/commonfiles/db-tasks.php

Die Voreinstellungen werden in `/db/message.php` gemacht.

Voreinstellung, welche Nachrichten aktiviert sind:

Note that if you change the values in `/db/message.php` and then upgrade the plugin the values will not automatically be changed in the config_plugins table where they are stored.

Der Versand der Nachrichten erfolgt über das Task-System. Das heißt der Nachrichten-Inhalt und die Bedingungen die zum Versand führen, sind in `/classes/task/trigger_mail.php` implementiert.

Neben den oben angeführten Voraussetzungen ist noch der Ablauf des Delays des Nutzers nötig, sowie seine Zustimmung zum Erhalt von Nachrichten. Diese Zustimmung ist in den Antworten in der Variable `choice_121` abgelegt.

---

1. Moodle: Cron-Job ↩

11. Aktivitätsabschluss

In einer Instanz der Lernstrategien, kann die Abschlussverfolgung auf automatisch gestellt werden. Dann werden dem Nutzer 8 ToDos angezeigt, die je nach erreichtem State als erledigt markiert werden.

11.1 Einstellung

Die Aktivitätenabschluss wird dann angezeigt, wenn die Abschlussverfolgung im Kurs (`Kurseinstellungen -> Abschlussverfolgung`) aktiviert ist (beide Einstellungen!) und sie in der Aktivität auch korrekt eingestellt ist:
`Drei-Punkte Menü der Lernstrategien-Aktivität -> Einstellungen -> Aktivitätsabschluss -> Abschlussverfolgung` : Abschluss, wenn alle Bedingungen erfüllt sind.

11.2 Implementation

Implementiert in `\classes\completion\` und `lib.php` .

Für die Settings wurde in `mod_form.php` die Funktionen `add_completion_rules()` und `completion_rule_enabled()` eingefügt.

Um den Status zu aktualisieren wird in `view.php` die Funktion `splernstrategien_view()` aufgerufen.

Ab Moodle 4.3 muss in den Formularen für die Completion ein Suffix eingebunden werden, das automatisch vergeben wird. Diese Funktione steht in 4.1 noch nicht zu verfügung.

11.3 Dokumentation

- Activity completion API
- Moodle Quellcode: mod_forum, mod_survey

12. Datenschutz-API

Moodle stellt zur Umsetzung der DSGVO eine Datenschutz-API (Privacy API) zur Verfügung.

Prinzipielle Hinweise zum Datenschutz erhält man auf den folgenden Seiten:

- Datenschutz in Moodle (allgemeine Hinweise)
- Datenschutz (für Administratoren)

Die Umsetzung der API erfolgt in `/classes/privacy/provider.php`. Es gibt eine vollständige Beschreibung der gespeicherten Daten und es werden Methoden zur Ansicht der gespeicherten Nutzendendaten und zur Löschung der Daten bereitgestellt.

Im Profil der Nutzenden ist es möglich, Datenschutzanfragen zu stellen.

13. Entwicklung

13.1 Updates durchführen

Wenn man eine neue Plugin-Version erzeugen will, muss man *mindestens* die folgenden Punkte beachten:

1. Release und Version in `version.php` anpassen.
   - Release folgt den SemVer-Regeln
   - Version ist aus dem Datum erzeugt: `YYYMMDDxx` mit `xx` als zweistelligem Zähler.
2. Datenbank-Definitionen anpassen.
   Neben eventuellen Strukturänderungen an der Datenbank sollte in der Datei `db/upgrade.php` immer der Savepoint für die aktuelle Version gesetzt werden.

   Wenn es Änderungen in den State-Definitionen gab, muss die entsprechende Tabelle gelöscht werden. Dies geschieht durch einen Eintrag in der `db/upgrade.php`:

```
if ($oldversion < YYYYMMDDxx) {
   ...
   // Clean States-table
   // Insert this snippet whenever there are changes to the
   // states-table (i.e. classes/states.php).
   // One occurrence on the last upgrade is sufficient.
   $tablename = 'splernstrategien_modules';
   $table = new xmldb_table($tablename);
   if ($dbman->table_exists($table)){
       $DB->delete_records($tablename);
   }
   ...
}
```

Wobei dieser Eintrag nur im letzten Versionssprung vorhanden sein muss.

13.2 Debugging

Um im Kurs Informationen zum aktuellen State und den bisherigen Nutzereingaben zu erhalten, muss man das Debugging in Moodle aktivieren. Dazu ruft man `admin/settings.php?section=debugging` auf und wählt in den Debug-Messages `DEVELOPER` aus und aktiviert `Display debug messages`.

Achtung: Der Debug-Modus zerstört das Layout!