

Handbuch zum XML-Backend

Christian Baumann

21. Juni 2010

Inhaltsverzeichnis

1	Voraussetzungen	3
2	Erstellen einer Aufgabe	3
2.1	Erstellen eines EC-Ordners	4
2.2	Erstellen einer Aufgabe	5
2.3	Das XML-Backend	5
2.3.1	Wohlgeformtheit	5
2.3.2	DTD Validation	6
2.3.3	XPath-Anfragen	6
3	Beispiele	7
3.1	Backend-Einstellungen	7
3.2	Einreichung	8
3.2.1	Nicht wohlgeformt	8
3.2.2	Nicht valide	8
3.2.3	Dokumentunterschiede	9
3.2.4	Fehlerfreie Einreichung	9

1 Voraussetzungen

Um das XML-Backend nutzen zu können, müssen folgende Voraussetzungen erfüllt sein:

- Plone
- ECAutoAssessmentBox
- ECSpooler
- XML-Backend

Nachdem alle Komponenten, wie in den beiliegenden Anleitungen, installiert wurden, können Sie das Backend verwenden, indem Sie unter Konfiguration → Auto Assessment Settings den verfügbaren (Available) Backends „XML (*Version*)“ den gewählten (Selected) hinzufügen (vgl. Abbildung 1).

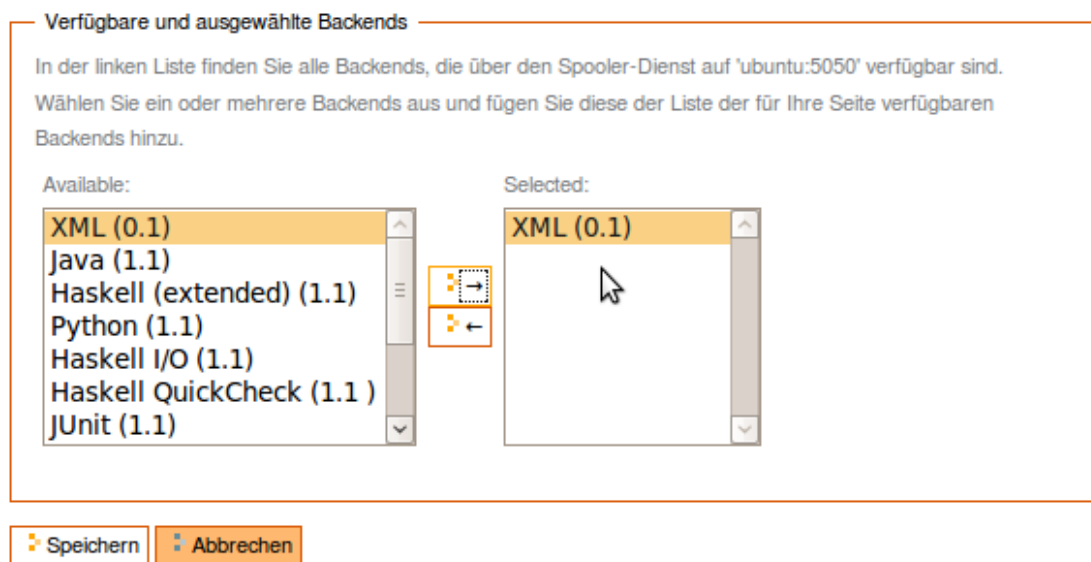


Abbildung 1: Hinzufügen des XML-Backends.

2 Erstellen einer Aufgabe

Im folgenden wird beschrieben, wie eine Aufgabe, die automatisch mit dem XML-Backend getestet werden soll, hinterlegt wird.

2.1 Erstellen eines EC-Ordners

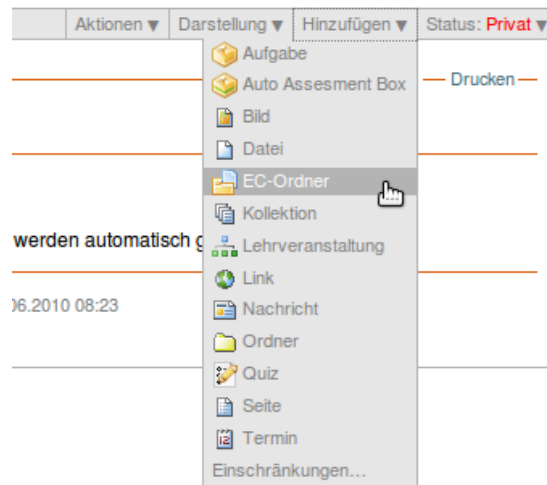


Abbildung 2: Anlegen eines EC-Ordners.

Aufgaben werden in Ordnern organisiert (vergleichbar mit einem Aufgabenblatt), die man wie folgt anlegt:

Auf der Plone-Startseite befindet sich das Menü „Hinzufügen“, in dem man den Punkt „EC-Ordner“ wählt.

Es wird ein Titel, gefolgt von einer kurzen Beschreibung und optionalen Hinweisen angegeben. Durch einen Klick auf „Speichern“ werden die Eingaben bestätigt und ein neuer Ordner angelegt, in den automatisch gewechselt wird.

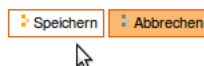


Abbildung 3: Speichern.

2.2 Erstellen einer Aufgabe

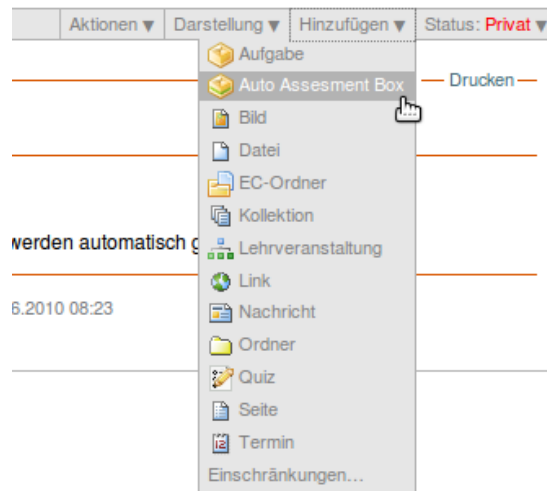


Abbildung 4: Hinzufügen einer Auto Assesment Box.

Zur Erstellung einer Aufgabe wählen Sie aus dem Menü „Hinzufügen“ den Punkt „Auto Assesment Box“ und füllen auf der erscheinenden Seite die Eingabefelder wie gefordert aus. Um das XML-Backend für diese Aufgabe nutzen zu können, wählen Sie unter dem Punkt „Backend“ „XML (*Version*)“ und bestätigen die Eingaben mit einem Klick auf „Speichern“.

2.3 Das XML-Backend

Klicken Sie nun auf „Bearbeiten“, gefolgt von „Backend“.

Das XML-Backend kann dazu genutzt werden, um Einreichungen von Studenten auf folgende Weisen zu testen:

- Wohlgeformtheit der Einreichung
- Validität der Einreichung anhand einer DTD
- Gleichheit von Ergebnissen von XPath-Anfragen

Hierzu sind einige Informationen im Backend zu hinterlegen:

2.3.1 Wohlgeformtheit

Um die Einreichung auf Wohlgeformtheit zu testen, aktiviert man die Checkbox „Test form“ (siehe Abbildung 5).



Abbildung 5: Checkbox zum aktivieren des Testens auf Wohlgeformtheit.

2.3.2 DTD Validation

Um eine studentische Einreichung gegen eine DTD validieren zu lassen, benötigt das Backend eine hinterlegte DTD, die Sie im Feld „DTD“ (vgl. Abbildung 6) angeben können.

Studenten steht es frei, ihre lokale DTD nach eigenem Wunsch zu benennen. Das XML-Backend erkennt den lokalen DTD Namen und ersetzt ihn mit der serverseitig gespeicherten DTD.

DTD
Enter a DTD the student's submission will be automatically validated against. If empty no validation will be performed.

Format (no change) ▼

Abbildung 6: Eingabefeld für eine DTD.

Beachten Sie, dass jede Eingabe in diesem Feld dazu führt, dass eine DTD Validation ausgeführt wird. Allein wenn dieses Feld leer ist, wird die DTD Validation übersprungen.

2.3.3 XPath-Anfragen

Das XML-Backend ist in der Lage XPath 2.0 Anfragen an eine studentische Einreichung zu stellen und diese mit einer Musterlösung zu vergleichen. Dazu hinterlegt man im Feld „Model solution“ (vgl. Abbildung 7) die Modelllösung und im Feld „XPath statements“ (vgl. Abbildung 8) zeilenweise XPath-Ausdrücke.

Model solution
If you want to run XPath statements on the student's submission, enter a model solution for this task. If no XML file is specified here, it will be ignored.

Format (no change) ▼

Abbildung 7: Eingabefeld für die Modelllösung.

Um als Lehrender die Dokumente in den XPath-Ausdrücken referenzieren zu können, wird die Variable „`{DOC}`“ verwendet, die intern in die tatsächlichen Dokumentennamen übersetzt wird.

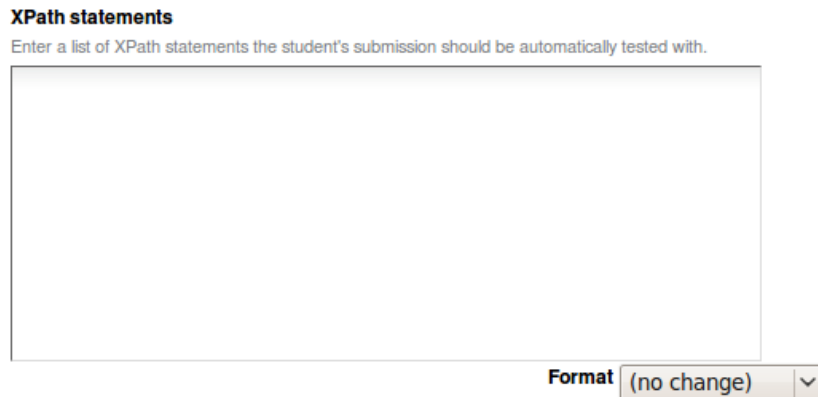


Abbildung 8: Eingabefeld für XPath Ausdrücke.

Sind sowohl eine Modelllösung als auch XPath-Ausdrücke angegeben, wird automatisch ein XQuery-Programm generiert, das die Anfragen an die Musterlösung und die studentische Einreichung stellt und die Ergebnisse dieser vergleicht.

3 Beispiele

Die folgenden Beispiele geben einen kurzen Einblick in die Funktionsweise des XML-Backends.

3.1 Backend-Einstellungen

Wir konfigurieren einen kompletten Test für dieses Beispiel, indem wir „Test form“ aktivieren, und folgende Texte in „DTD“, „Model solution“ und „XPath statements“ hinterlegen.

```
<!ELEMENT hallo (#PCDATA)>
```

Listing 1: Beispiel DTD.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "dtd.dtd">
<hallo>Hallo Welt!</hallo>
```

Listing 2: Modelllösung.

```
count({DOC})
${DOC}/hallo
```

Listing 3: Beispiel XPath-Ausdrücke.

3.2 Einreichung

3.2.1 Nicht wohlgeformt

Die Einreichung

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hallo>Hallo Welt!</hallo
```

Listing 4: Nicht wohlgeformte Einreichung.

führt zu der in Abbildung 9 dargestellten Rückmeldung.

Antwort:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hallo>Hallo Welt!</hallo
```

 [sherry.20100611.233531.txt](#) — Plain Text, 0Kb

Automatische Rückmeldung:

```
Your submission failed. It is not well-formed.

Recieved result: submission.xml:4: parser error : expected '>'
^
```

Abbildung 9: Rückmeldung einer nicht wohlgeformten Einreichung.

3.2.2 Nicht valide

Die Einreichung

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hola>Hallo Welt!</hola>
```

Listing 5: Nicht valide Einreichung.

führt zu der in Abbildung 10 dargestellten Rückmeldung.

Antwort:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hola>Hallo Welt!</hola>
```

[sherry.20100611.233707.txt](#) — Plain Text, 0Kb

Automatische Rückmeldung:

```
Your submission failed. It is not valid.

Recieved result: submission.xml:3: element hola: validity error : No declaration for element hola
Document submission.xml does not validate against dtd.dtd
```

Abbildung 10: Rückmeldung einer nicht validen Einreichung.

3.2.3 Dokumentunterschiede

Die Einreichung

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hallo>Guten Tag Welt!</hallo>
```

Listing 6: Beispiel für eine semantisch falsche XML-Datei.

ist wohlgeformt und lässt sich gegen die hinterlegte DTD validieren. Allerdings sind die Textdaten innerhalb des hallo-Tags in der Einreichung von denen in der Modelllösung verschieden, was zu einer in Abbildung 11 dargestellten Rückmeldung führt.

Antwort:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hallo>Guten Tag Welt!</hallo>
```

[john.20100613.143923.txt](#) — Plain Text, 0Kb

Automatische Rückmeldung:

```
Your submission failed. A XPath expression evaluated against a sample solution and compared to your submission yielded a different result.
Test case was:
/hallo
```

Abbildung 11: Rückmeldung bei Unterschieden in den verglichenen XML-Dateien nach einer XPath-Anfrage.

3.2.4 Fehlerfreie Einreichung

Die fehlerfreie Einreichung

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hallo>Hallo Welt!</hallo>
```

Listing 7: Fehlerfreie Einreichung

führt zu der in Abbildung 12 dargestellten Rückmeldung.

Antwort:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE hallo SYSTEM "myLocal.dtd">
<hallo>Hallo Welt!</hallo>
```

 john.20100613.164946.txt — Plain Text, 0Kb

Automatische Rückmeldung:

```
Your submission is well-formed.
Your submission was successfully validated.
Your submission was successfully tested.
```

Abbildung 12: Rückmeldung im Falle einer fehlerfreien Einreichung.