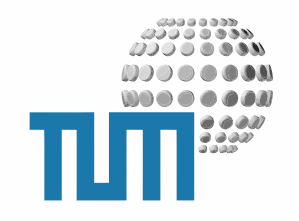


RemoteXMLObject

Content Managers Manual

Einbindung von XML-Daten externer Systeme

Version 1.0
preliminary

TUM WWW & Online Services
Richard-Wagner-Strasse 18
D-80333 München

E-Mail: info@ze.tum.de

Entwicklerteam der Technischen Universität München:

Dr. rer.-nat. Thomas Wagner (Wissenschaftliche Leitung)
Dipl.-Ing. Thomas Mehlhart
Gerhard Schmidt
Christian Hamm

Inhaltsverzeichnis

1 Allgemeines

RemoteXMLObjects erlauben es, von externen Servern bereitgestellte XML-Dateien zu parsen und im Portal-Kontext als Object Model abzubilden auf das mit den üblichen Methoden von Zope/myTUM zugegriffen werden kann und das in den Portalkontext eingebunden ist. Beim Abruf des XML können Parameter übergeben werden. Für Fehler beim Abruf des XML bzw. Nichterreichbarkeit des externen Servers steht eine Fehlerbehandlung sowie eine Caching-Funktion zur Verfügung.

Auf das Objekt-Modell kann mittels Python oder Zope PageTemplate Code zugegriffen werden. Dabei stehen auch die üblichen Hilfsmethoden des myTUM-Systems zur Verfügung. Die Darstellung kann durch einen internen oder beliebige externe Views erfolgen. Elemente, Elementattribute oder Elementwerte des Objektmodells können beliebig im Zope/myTUM-Kontext genutzt werden. Der Zugriff auf die Elemente des XML erfolgt durch XPath-Expressions die über eine Python-API oder in TALES-Pathexpressions eingesetzt werden können.

RemoteXMLObjects sind mehrsprachig, d.h. es kann für jede vom Portal unterstützte Sprache ein eigenes XML abgerufen und durch ein sprachenspezifisches PageTemplate dargestellt werden.

Der gerenderte Inhalt des XML wird als Volltext im Portalcatalog indiziert.

2 RemoteXML-Objekt anlegen und konfigurieren

Für jedes abzurufende XML müssen Sie ein eigenes RemoteXMLObject anlegen. Allerdings können diesem ggf. Parameter übergeben werden, die dann beim Abrufen des XML vom RemoteServer als Querystring mitgegeben werden. Das Anlegen erfolgt wie gewohnt über die Inhaltssicht und Auswahl des entsprechenden Objekttyps. Nach dem möglicherweise erforderlichen Ausfüllen des Metadaten-Formulars können Sie das RemoteXMLObject konfigurieren.

2.1 Allgemeine Angaben

Bei den allgemeinen Angaben zum RemoteXML-Object können Sie einen Titel und eine Beschreibung für das Objekt angeben. Beide Angaben werden bei der Suche berücksichtigt und sollen dem Anwender helfen, den zu erwartenden Inhalt des Objekts zu erkennen. Sowohl Titel als auch Beschreibung sind mehrsprachig, d.h. für jede vom Portal unterstützte Sprache können unabhängig Titel und Beschreibung festgelegt werden.

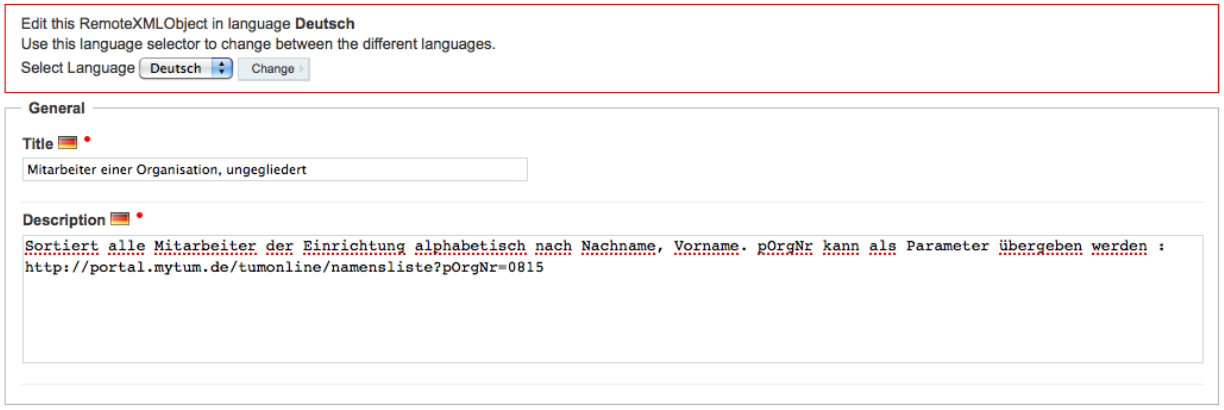


Abbildung 1: Sprachauswahl, Titel und Beschreibung

2.2 Einstellungen zur XML-Quelle

Hier nehmen Sie die Einstellungen für den Abruf des XML vom Server vor. Für jede Sprache die vom Portal unterstützt wird können Sie eine eigene URL angeben, sodass sprachabhängig unterschiedliche XMLs abgerufen werden können. Zwar kann in der Regel die Sprache auch als Parameter an, bzw. durch das RemoteXML-Objekt übergeben werden, dann kann allerdings kein Caching eingesetzt werden. Wenn Sie stattdessen für jede Sprache eine eigene URL angeben, werden die Daten gechached, sofern keine anderen Parameter verwendet werden:

```
https://campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=de&pOrgNr=26134
```

bzw. für englische Abfrage:

```
https://campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=en&pOrgNr=26134
```

Wird nur eine Target-URL angegeben, so wird diese für alle Sprachen verwendet.

Bei 'Update-Interval' geben Sie das Zeitintervall an, indem die URL abgerufen und das DOM upgedated werden soll. Die Einstellung gilt grundsätzlich für alle Sprachen, d.h. bei der Angabe verschiedener URLs werden die URLs aller Sprachen gleichzeitig abgerufen. Unterschiedliche Updatezeiten für die einzelnen Sprachen sind also nicht möglich.

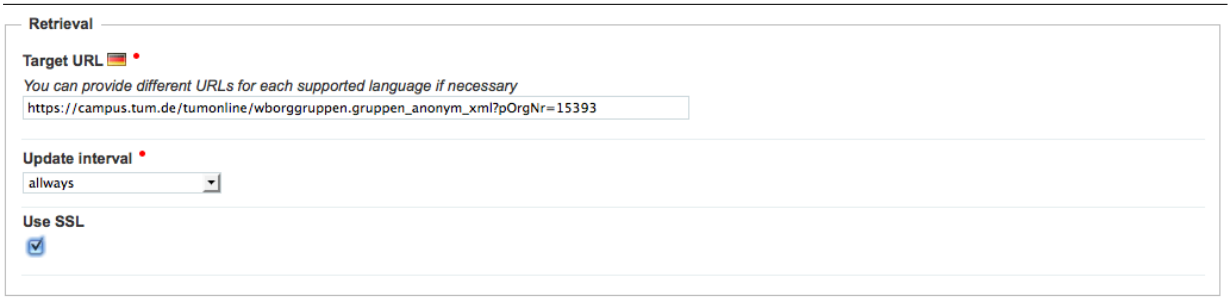


Abbildung 2: Einstellungen zum XML-Abruf

Beachten Sie bitte, dass ein Caching nur möglich ist d.h. eine Intervallzeit nur sinnvoll ist, wenn der Abruf der URL ohne dynamische Parameter erfolgt. Wird ein Parameter übergeben, entweder manuell oder durch die Parameter-Einstellungen, wird das XML immer aktuell abgerufen, da sonst die Ergebnisse der letzten Query verwendet werden würden. Daher wird bei eingestelltem Parameter-Mapping (s.u.) und tatsächlich gesetzten Parametern immer bei jedem Aufruf des RemoteXMLObject ein Update durchgeführt, unabhängig von den hier vorgenommenen Intervalleinstellungen.

Schließlich müssen Sie noch angeben, ob der Abruf des XML über SSL-Protokoll erfolgen soll. Diese Einstellung steuert die Protokolle 'http' oder 'https' aus, unabhängig davon, welches Protokoll in der Target-URL angegeben wurde.

2.3 Parameter Mapping

Grundsätzlich können Sie dem RemoteXML auch Parameter übergeben, die dann als Querystring an den Remote-Server weitergereicht werden. Die Übergabe der Parameter

kann durch den Request erfolgen (also durch Aufruf des RemoteXML-Objekts mit Querystring), durch Aufruf der Methode getDOM() mit entsprechenden Keywordparametern, z.B. getDOM(language='de') aber auch automatisch bei Aufruf des RemoteXML-Objekts. Dadurch können beliebige Daten aus dem Portal als Parameter zur Laufzeit an den Remote-Server übergeben werden.

Grundsätzlich werden nur Parameter übergeben die im Parameter-Abschnitt konfiguriert werden. Diese Parameter werden bei Aufruf des remoteXML (oder dem Call von getDOM()) aus dem Request oder den Keyword Arguments extrahiert oder auf den Wert der entsprechend aufgerufenen Methode im Portal gesetzt. Der Wert wird immer als String übergeben. Parameter ohne Value (Leerstring, None) werden ignoriert.

Die einzelnen Parameter geben Sie zeilenweise ein. Beachten Sie beim Parameternamen bitte die Gross- und Kleinschreibung sowie ggf. die Anforderungen des Remote-Systems.

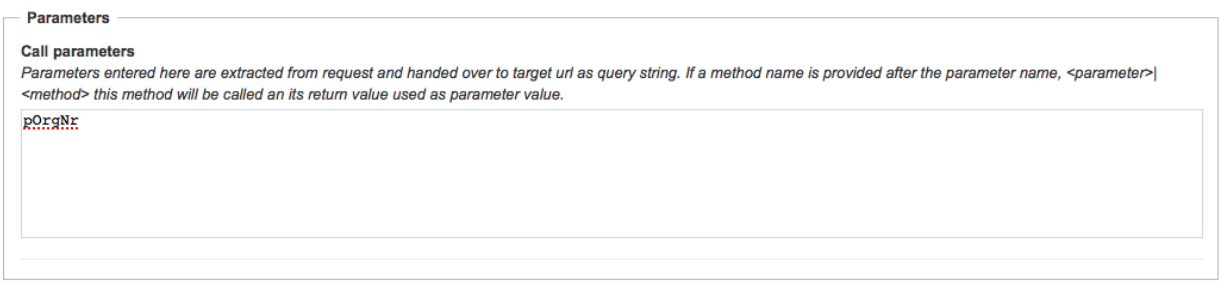


Abbildung 3: Einstellung der auszuwertenden Parameter

Wenn Sie hinter den Parameternamen durch einen '|' getrennt einen TALES-Ausdruck eingeben, wird dieser beim Aufruf des RemoteXML evaluiert und der Rückgabewert als Wert für den angegebene Parameter gesetzt. Alle in diesem TALES-Ausdruck verwendeten Objekte und Methoden müssen im context aufrufbar sein, der Rückgabewert sollte ein Text sein. Anderenfalls wird die String-Repräsentation des Returnwerts verwendet. Achten Sie auch darauf, dass sichergestellt ist, dass das RemoteXML aufrufende Benutzer auch alle erforderlichen Berechtigungen haben um den Ausdruck auszuführen.

Sind Parameter gesetzt und werden beim Aufruf tatsächlich Parameter übergeben oder automatisch gesetzt, so wird das XML grundsätzlich neu abgerufen. Die gesetzten Parameter werden als Querystring an den Remote-Server übergeben. Wurden in der Target-URL bereits Query-Parameter definiert, so werden diese durch die ggf. übergebenen Werte ersetzt oder wenn keine Werte für diese Parameter übergeben wurden oder die Parameter nicht definiert wurden entsprechend beibehalten und die übergebenen Parameter ergänzt.

Angenommen Sie haben ein RemoteMLObjekt angelegt unter der URL:

```
http://portal.mytum.de/tumonline/xmltest
```

Dort haben Sie als Target-URL konfiguriert:

```
https://campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=en&pOrgNr=26134
```

Ausserdem haben Sie in der Parameter-Sektion des RemoteXML folgende Parameter definiert:

```
pOrgNr
language
```

Werden keine Parameter übergeben, so wird die URL wie angegeben aufgerufen. Rufen Sie aber die URL des RemoteXMLObjects mit einen der angebenen Parameter auf, z.B.

```
?pOrgNr=15393
```

so wird an Stelle der bei Target-URL angegebenen URL die URL wie folgt aufgerufen:

```
https:,//campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=en&pOrgNr=15393
```

Übergeben Sie zusätzlich noch den Parameter language, z.B.

```
?pOrgNr=15394&language=de
```

so wird beim Aufruf der URL der Parameter 'language' durch den entsprechenden Wert ersetzt:

```
https://campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=de&pOrgNr=15393
```

Werden im Querystring Parameter übergeben, die nicht in der Parameter-Sektion konfiguriert wurden, so stehen diese zwar wie gewohnt über den Request im Viewtemplate des RemoteXMLObjects zur Verfügung, diese werden jedoch beim Abruf des XML nicht an den Remote-Server weitergereicht, d.h. bei einem Querystring

```
?pOrgNr=15394&language=de&status=unklar
```

wird ebenfalls nur die folgende URL vom Remote-Server abgerufen:

```
https://campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=de&pOrgNr=15393
```

Nun ist es des öfteren notwendig oder wünschenswert, Daten aus dem Portalcontext dynamisch an den externen Server zu übertragen, z.B. die Benutzerkennung des angemeldeten Benutzers oder dessen Organisationseinheit. Deshalb können die Parameter der vom Remote-Server abgerufenen URL auch durch TALES-Expressions definiert werden:

Dazu geben Sie in der Parameter-Sektion der Konfiguration hiner dem Parameter mit '|' getrennt den TALES-Ausdruck an, dessen Rückgabewert für den Parameterwert verwendet werden soll:

```
pOrgNr|python:member.getProperty('tumUniviskey')
language|context/portal_languages/getPreferredLanguage
```

Als Bindings stehen 'portal', 'request', 'object', 'context', 'member', 'nothing' in der üblichen Definition zur Verfügung. Es können sowohl Path- als auch Python-Expressions verwendet werden.

Im vorliegenden Beispiel wird bei Aufruf des RemoteXMLObjekts zunächst der Organisationsschlüssel ermittelt (trotz der Bezeichnung 'univiskey' die aktuelle Org-Nr in TUMOnline) und die aktuell vom Benutzer gewählte Spracheinstellung. Dann werden diese Daten via Querystring an den Remote-Server übergeben, z.B.

```
https://campus.tum.de/tumonline/wborggruppen.gruppen_anonym_xml
?language=af&pOrgNr=TU00000.TUZVFZV.TUZVZA7.TUZVR7O
```

Parameter können aber auch bei Abruf des DOM übergeben werden. Dazu kann auf dem RemoteXML-Objekt die Methode getDOM() mit den entsprechenden Parametern als Keyword-Parameter aufgerufen werden.

```
context.getDOM(language='de', pOrgNr='12345')
```

Auch hier wird die URL vom Remote-Server mit den entsprechenden Parametern aufgerufen.

Ebenfalls möglich ist auch der Aufruf des RemoteXML-Objects mit diesen Parametern. Wenn also das RemoteXLM-Objekt im Context unter der ID 'testobjekt' existiert, dann kann man es von einem anderen Script, Template aufrufen durch

```
context.testobjekt(pOrgNr='12345')
```

Bitte beachten Sie, dass in jedem Fall bei einem angebenen Parameter-Mapping grundsätzlich das DOM bei jedem Aufruf neu generiert wird, unabhängig davon, was als Update-Intervall eingestellt wurde. Sind Parameter definiert, erscheint im Konfigurationsformular bei den Update-Einstellungen ein entsprechender Hinweis.

2.4 Darstellung

Zunächst dient ein RemoteXML-Objekt nur dazu, die übergebenen Parameter an den Remote-Server zu übergeben und das als Antwort erhaltene XML in eine Zope Objektstruktur umzuwandeln, die dann mit einer beliebigen Präsentationslogik dargestellt werden kann (PDF, LaTeX, PageTemplate, XSLT).

Der Administrator kann im Abschnitt 'Darstellung' festlegen, welche Präsentationsmethode bzw. welcher View für das RemoteXML als Standardansicht verwendet werden soll. Selbstverständlich können aber auch beliebige Views explizit aufgerufen werden. Sie können also als normale Ansicht die Interne Ansicht mit Decoration verwenden, aber natürlich trotzdem für explizite Fälle das XML mit dem View 'remotexml_raw_view' aufrufen um dann eine dekorationsfreie Darstellung zu bekommen.

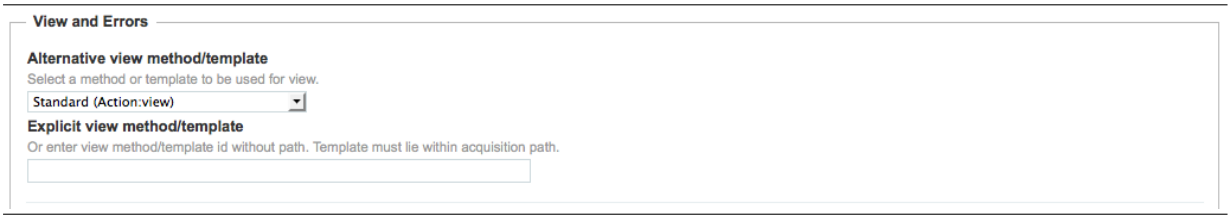


Abbildung 4: Darstellungsoptionen

Die einfachste Form der Darstellung ist die Verwendung des eingebauten Views, der Zope PageTemplate-Code verwendet und die Verwendung von Path- bzw. Python-TALES-Ausdrücken sowohl für XML-Elemente als auch Portalobjekte erlaubt und vollständig in das Portal integriert ist. Das bedeutet, dass der View immer Zugriff auf den gesamten Portalkontext hat und alle Möglichkeiten normaler ZPT-Dokumente bietet. Hier kann zwischen einer Darstellung mit Decoration und einem Raw-View gewählt werden.

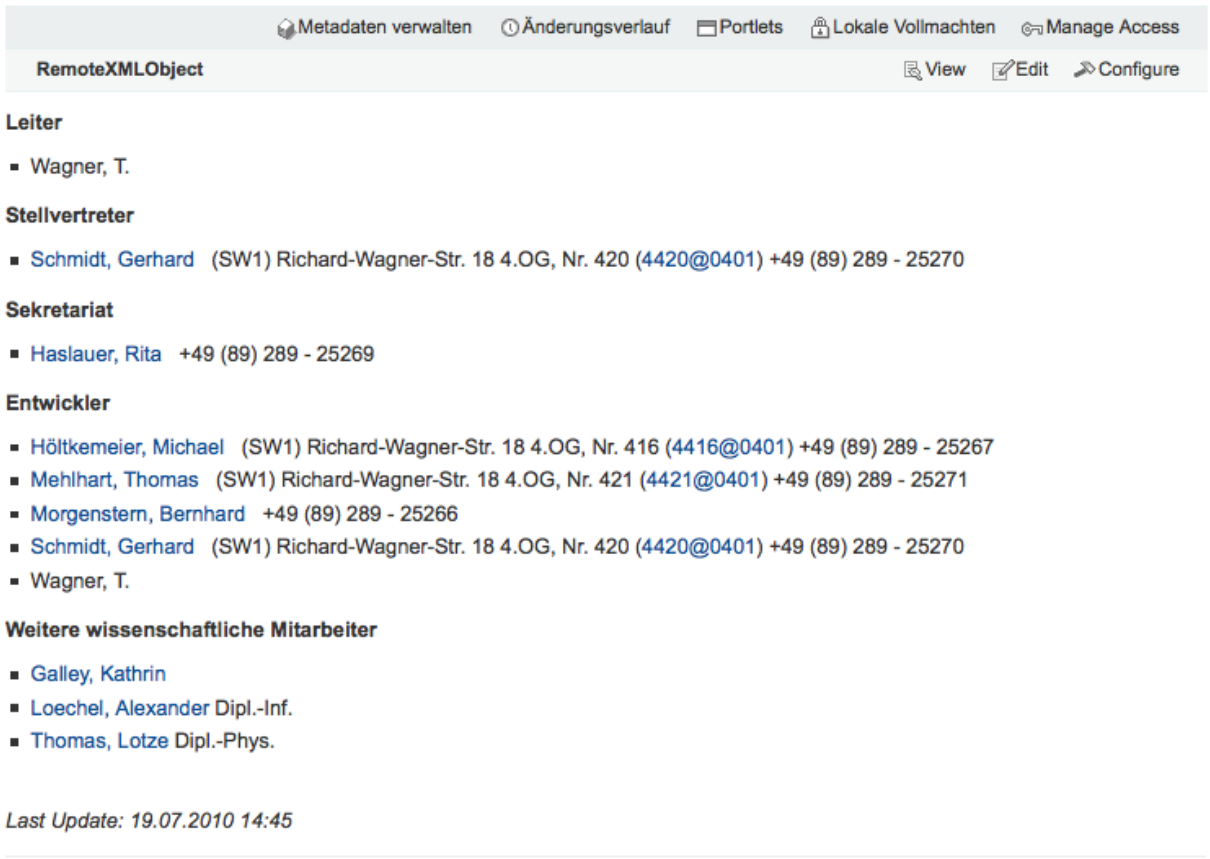


Abbildung 5: Darstellung innerhalb der Decoration

In ersterem Fall wird das RemoteXML m Portaldesign mit Header, Portlets usw. gerendert, während beim raw-View nur der eigentliche Inhalt des Views dargestellt wird. Dadurch kann dann das RemoteXML in anderen Systemen als HTM-Fragment eingebaut werden. Normalerweise sollten Sie für die Darstellung den internen View auf Zope Pagetemplate Basis mit Decoration wählen. Für den Fall, dass das gerenderte XML in anderen Webserven als HTML-Fragment verwendet werden soll, können Sie (ggf. explizit) den Raw-View verwenden.

Leiter

- Wagner, T.

Stellvertreter

- Schmidt, Gerhard (SW1) Richard-Wagner-Str. 18 4.OG, Nr. 420 (4420@0401) +49 (89) 289 - 25270

Sekretariat

- Haslauer, Rita +49 (89) 289 - 25269

Entwickler

- Höltkemeier, Michael (SW1) Richard-Wagner-Str. 18 4.OG, Nr. 416 (4416@0401) +49 (89) 289 - 25267
- Mehlhart, Thomas (SW1) Richard-Wagner-Str. 18 4.OG, Nr. 421 (4421@0401) +49 (89) 289 - 25271
- Morgenstern, Bernhard +49 (89) 289 - 25266
- Schmidt, Gerhard (SW1) Richard-Wagner-Str. 18 4.OG, Nr. 420 (4420@0401) +49 (89) 289 - 25270
- Wagner, T.

Weitere wissenschaftliche Mitarbeiter

- Galley, Kathrin
- Loechel, Alexander Dipl.-Inf.
- Thomas, Lotze Dipl.-Phys.

Abbildung 6: Darstellung als Raw-View

Weitere Views sind eine Source-Ansicht, bei welcher das XML als solches präsentiert wird, die DOM-Ansicht, die dem berechtigten Benutzer ein Browsen im DOM des XML erlaubt sowie eine einfache Statistikansicht. Diese Standardansichten sind überwiegend für administrative Zwecke gedacht, wenn das RemoteXMLObject nicht direkt als Content eingesetzt wird, sondern programmatisch bzw. als Proxy-Service eingesetzt wird.

Der Einsatz von XSLT zum Rendern des XML ist ebenfalls möglich, wird jedoch nicht empfohlen. Hier findet keine Integration in den Portal-Kontext statt, d.h. Ihnen stehen keine Hilfsmethoden zur Verfügung, es besteht kein Zugriff auf die Portalobjekte und die Decoration kann nicht eingebaut werden. Der Einsatz von XSLT ist nur sinnvoll, wenn das RemoteXMLObject nicht als Contenobjet sondern als reines HTML-Fragment eingesetzt werden soll.

Sie können das RemoteXML aber auch als iCal, CSV oder anderes sinnvolles Format ausgeben lassen. Dazu benötigen Sie ein eigenes Pythonscript, Pagetemplate oder eine andere Skinmethode für die Darstellung bzw. Ausgabe. Dazu müssen Sie diese als explizite externe Methode oder Viewtemplate angeben, das dann beim Aufruf für die Darstellung verwendet wird. Die Methode muss im Acquisitionspfad liegen und die das RemoteXMLObject aufrufenden Benutzer müssen für diese Methode auch berechtigt sein.

Schließlich können RemoteXMLObjekts auch im PersonalWorkplace oder iGoogle als Gadget dargestellt werden. Dazu wird als Gadget-URL einfach das RemoteXML-Object einfach mit dem View 'remotexml_gadgetview' aufgerufen.

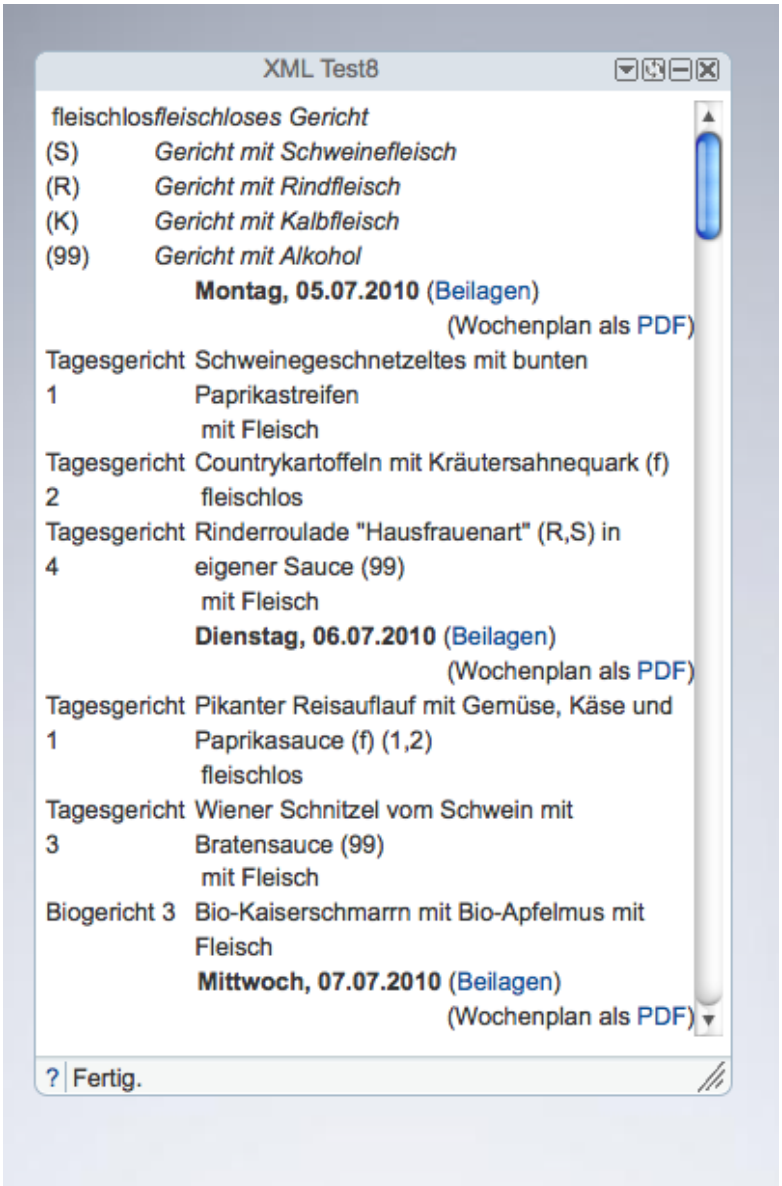


Abbildung 7: Quick & Dirty Darstellung des RemoteXML als Gadget

Bei entsprechender Konfiguration der Parameter (siehe 'Parameter Mapping') kann der Benutzer diese über die Einstellungen des Gadgets konfigurieren.

2.5 Fehlerbehandlung

Schließlich legen Sie in diesem Abschnitt auch fest, wie sich das RemoteXMLObject im Fehlerfall verhalten soll. Mögliche Fehler sind die Nichtverfügbarkeit des Remote-Servers oder die Auslieferung eines ungütigen XML.

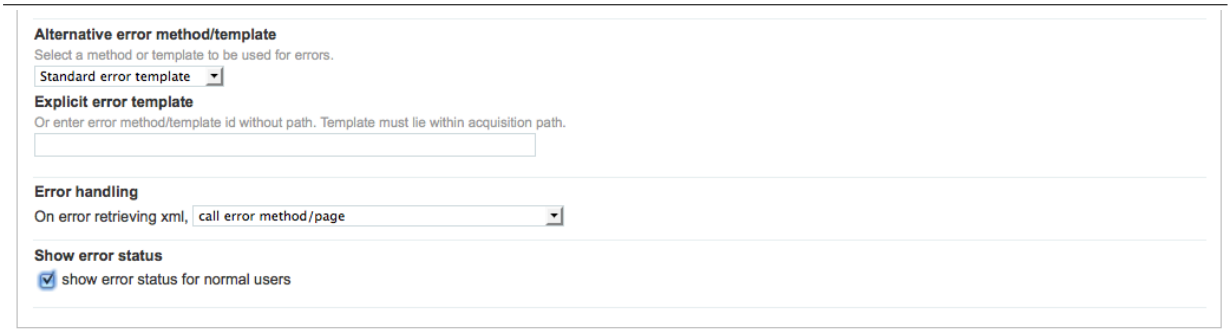


Abbildung 8: Einstellungen zur Fehlerbehandlung

Hier geben Sie an, welche Methode bzw. welches Template für die Behandlung eines Fehlers aufgerufen wird. Damit können Sie eigene Fehlerseiten definieren, die z.B. einen spezifischen Ansprechpartner anzeigen oder eine Methode bzw. komplexe Prozesse zur Fehlerbehandlung aufrufen (Workflows, Scripts usw.)

Nun stellen Sie noch ein, wann und wie im Fehlerfall zu reagieren ist. Dabei können Sie angeben, ob die Fehlermethode sofort, erst nach einer bestimmten Zeitspanne oder gar nicht aufgerufen werden soll. Als Tsandrad-Fehlermeldung erhält der Benutzer eienn Hinweis, dass ein Abrufen des XML nicht möglich war und er die Seite zu einem späteren Zeitpunkt nochmal aufrufen soll.

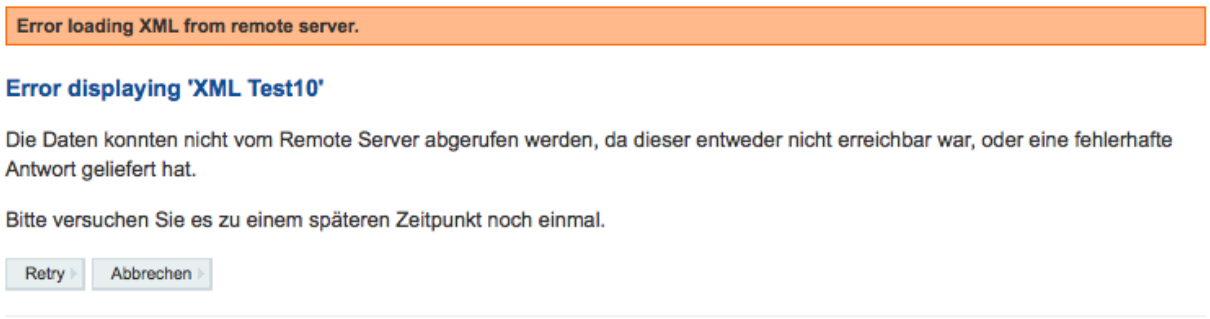


Abbildung 9: Hinweis, dass XML nicht abgerufen werden konnte

Für den Fall, dass die Fehlermethode erst verzögert oder gar nicht aufgerufen wird, werden die Daten des XML bzw. DOM bis zum Beheben des Fehlers aus dem Cache des RemoteXMLObjects verwendet. Beachten Sie hierbei, dass bei parameterisierten RemoteXMLObject, also für den Fall, dass Argumente übergeben werden, immer sofort die

Fehlermethode aufgerufen wird, da ein Caching hier nicht möglich ist. Sind entsprechende Parameter konfiguriert, wird in diesem Abschnitt analog zum Updateintervall ein entsprechender Hinweis eingeblendet.

Abbildung 10: Hinweis, dass XML nicht abgerufen werden konnte und gecachte Daten verwendet werden

Abschließend geben Sie noch an, ob der normale Benutzer im Fehlerfall per Status-Message einen Hinweis bekommen soll, dass die dargestellten Daten aus dem Cache stammen. Benutzer mit Administrationsberechtigung für das RemoteXMLObject erhalten diese Meldung immer unabhängig von der Einstellung.

Fehler bei der Erstellung des Viewtemplates werden durch die Fehlebehandlung nicht hier abgefangen, sondern durch den üblichen Mechanismus des Formcontrollers.

2.6 Cataloging

Wenn gewünscht, kann das gerenderte XML mit seinen dargestellten Inhalten im Portal-Catalog indiziert werden. Ausserdem können einzelne oder aggregierte Werte von XML-Elementen auf bestimmte Indices gemapped werden. Dadurch kann das RemoteXML auch bei der Suche gefunden werden.

Im Abschnitt 'Cataloging' legen Sie dazu fest, ob die gerenderten Daten des RemoteXMLObjects generell im Portalcatalog katalogisiert werden sollen. Wenn diese Option nicht aktiviert ist, werden nur Titel und Description katalogisiert, das gerenderte XML aber ignoriert. Beachten Sie bitte, dass ein Cataloging der XML-Daten nur sinnvoll ist, wenn es sich um quasi statische Seiten handelt, also keine Parameter bzw. feste Parameter übergeben werden (z.B. Seite mit Anzeige der Mitarbeiter einer festen Organisationseinheit).

Sie können darüberhinaus festlegen, ob und welche Daten des XML auf welchen Index des PortalCatalogs gemapped werden soll. Die Eingabe erfolgt zeilenweise im Format 'indexname:xpath', z.B.

```
keywords://keyword
```

Dem Index werden dann die textuellen Werte aller Tags mit Name 'keyword' im aktuellen DOM übergeben. Wird nur ein Tag gefunden, wird der Wert als Text an den Index

übergeben, werden über den XPath mehrere Elemente gefunden, werden die Werte als Liste übergeben.

Im oben dargestellten Fall werden für das folgende XML:

```
<metadata>
    <keyword>hurz</keyword>
    <keyword>blah</keyword>
    <keyword>fasel</keyword>
</metadata>
```

im Index 'keywords' die Stichworte ['hurz', 'blah', 'fasel'] indiziert. Es ist Aufgabe des Administrators darauf zu achten, dass die gemappten Elementwerte und der gewünschte Index korrespondieren, allerdings werden folgende Annahmen getroffen: bei einem Keyowrdindex werden die Werte immer als Liste übergeben und bei einem Date-Index wird um den Rückgabewert (bei mehreren Werten wird nur der erste genommen) ein DateTime ausgeführt. Tritt beim Indizieren ein Fehler auf, wird der Wert ignoriert.

Generell macht eine Indizierung der XML-Daten im Katalog nur Sinn, wenn diese Daten nicht personalisiert bzw. parameterisiert ermittelt werden. Ausserden wird vorausgesetzt, dass das XML-Schema für jeder der eingesetzten Sprachen gleich ist, d.h. sich die Tags der möglicherweise unterschiedlichen Sprachversionen nicht unterscheiden, sondern nur Unterschiede im textuellen Inhalt der Tags sind.

3 Erstellen eines internen Views für das RemoteXML

Wie bereits unter 'Configuration' erwähnt, können RemotXML mit beliebigen Präsentationsmethoden dargestellt werden. Die wichtigste Methode ist das eingebaute View-Template, das sowohl mit, als auch ohne Decoration angezeigt werden kann. Neben der Verwendung von Standard ZPT/TALES sowohl für Portalobjekte als auch für XML-Elemente, bietet dieser View die vollständige Integration in den Portalcontext und alle notwendigen Bindings.

Das geparste XML steht innerhalb des RemoteXMLObjects als Objekstruktur zur Verfügung. Diese kann dann genauso wie Portalobjekte mittels TALES Pathexpressions oder TALES Pythonexpressions abgefragt und mittels Zope Page Template Code gerendert werden. Das Root-Objekt des XML steht als Binding 'dom' (Domain Object Modell) zur Verfügung, die einzelnen Elemente des XML werden als RemoteXMLElement dargestellt. Bei XML-Elementen erfolgt die Traversierunge bei Pfadausdrücken durch XPATH (siehe auch Kapitel XPath) oder durch die API der XML-Objekte und des DOMs.

3.1 Internes View-Template für das XML anlegen

Um ein internes View-Template zu erstellen, rufen Sie das RemoteXMLObject auf und wählen die Objektaktion 'Edit'. Sie können nun, gegebenenfalls für jede der unterstützen Sprachen getrennt, ein Pagetemplate definieren, das die Darstellung des XML übernimmt. Da Sie sich dabei im Zope/Portal-Context befinden, haben Sie Zugriff auf alle Objekte, Tools und Methoden des Portals. Der Zugriff auf das DOM des XML erfolgt durch ein spezielles Binding 'dom' das zusätzlich zu den Standardbindings angeboten wird. Im Pagetemplate können beliebige TALES Pathexpressions oder Python-Expressions verwendet werden. Der Zugriff auf die Elemente des DOM und deren Werte erfolgt durch XPath bzw. modifizierte XPath-Expressions. Bei Pfadausdrücken wird via XPath eine Traversierung durch das DOM simuliert.

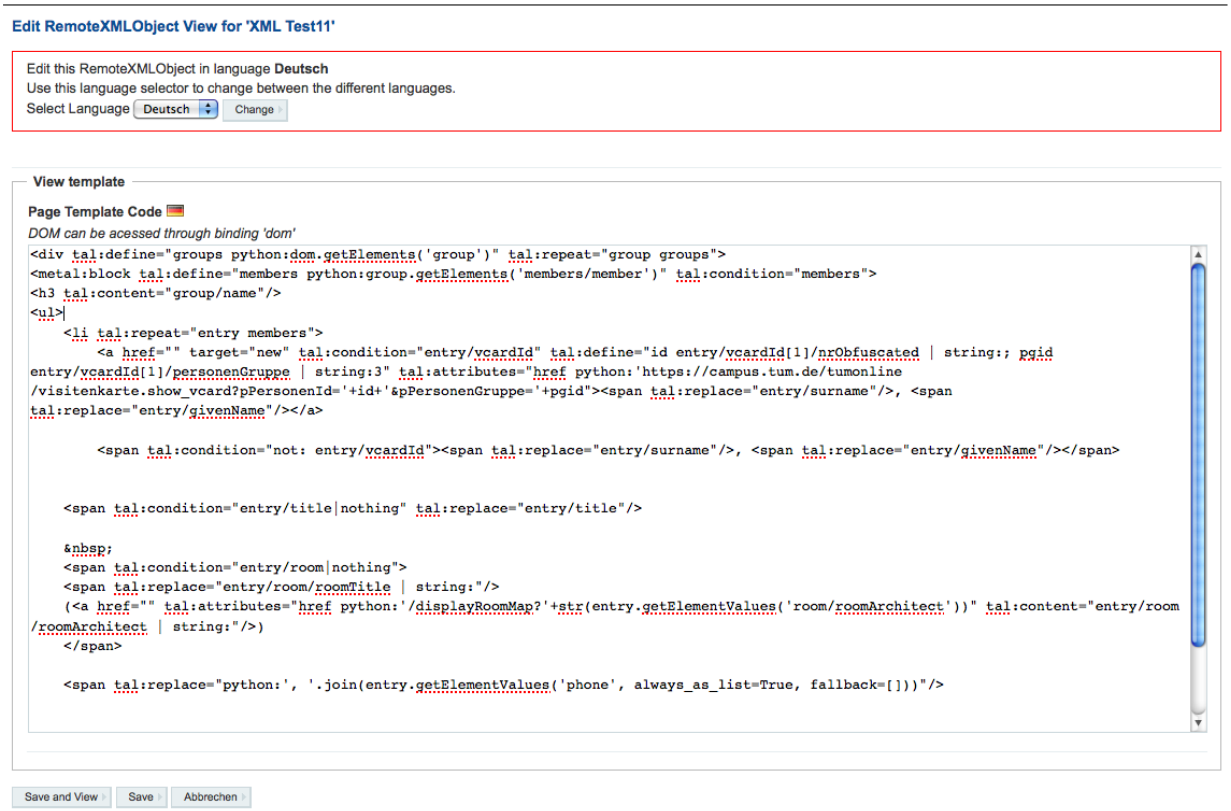


Abbildung 11: Bearbeiten des Views

3.1.1 Verwendung von Path-Expressions

Solange Pathexpressions auf normale Portalobjekte angewendet werden bestehen keinerlei Unterschiede zu normalen Page Templates. Bei der Verwendung von Pfadausdrücken auf das Binding 'dom' bzw. auf entsprechende DOM Elemente des geparsten XML wird bei Pfadausdrücken eine Traversierung durch das DOM via XPath simuliert. Der Pfadausdruck 'domcoursecourseID' entspricht nicht einem Zope-Objektpfad auf ein (per Definition immer eindeutiges) Objekt, sondern der Ausdruck liefert das Ergebnis einer XPath-Query auf dem im XML-Root mit dem XPath-Ausdruck 'coursecourseID'. Das kann im Spezialfall ein Element sein, es können aber auch mehrere Elemente sein.

Um die Handhabung der XML-Objekte unter TALES zu vereinfachen, gilt dabei für die Pathexpressions die folgende Regel:

Existiert zu der XPath-Query ('coursecourseID') genau ein XML-Element, so wird dessen Textueller Wert (der Inhalt zwischen dem enstprechenden Start- und Endtag) verwendet. Liefert die XPath-Query eine Liste von Elementen zurück, muss diese entsprechend in TALES eingebunden werden.

Wird kein passendes Element gefunden, wird ein None zurückgegeben, das ebenfalls

ggf. abgefangen werden muss.

Da die XPath-Spezifikation (z.B. Queries mit '' oder '::') mit TALES kollidiert werden müssen die entsprechenden, diese Zeichen enthaltenden XPath-Ausdrücke in TALES-Pfadexpressions maskiert werden.

Anstelle von '' wird dabei '##' eingesetzt, statt einem '::' die Kombination '++', also statt dem regulären XPath-Ausdruck 'courseID' muss in Pfadausdrücken '##courseID' verwendet werden.

Generell ist die Verwendung von Pfad-Ausdrücken bequem und schnell. Allerdings hat diese Notation (wie auch bei ihrer normalen Verwendung bei ZPT) klare Limitierungen. Sei es die fehlende Möglichkeit, Parameter zu übergeben oder die Problematik mit verschiedenen Types umzugehen. Für solche Fälle muss auf TALES-Pythonexpressions und die API der XML-Elemente zurückgegriffen werden.

3.1.2 Python-Expressions

Die Verwendung von Python-Expressions ist zwar etwas aufwendiger aber generell unproblematisch und konsistenter. Sie unterscheidet sich nicht von der Verwendung in normalen Zope Page Templates. Die Anwendung kann direkt auf dem Binding 'dom' erfolgen. Das Binding 'here' bzw. 'context' bezieht sich auf das RemoteXMLObject selbst, nicht auf das DOM des durch dieses Objekt verwalteten XML. Wenn dem DOM Parameter übergeben werden sollen, muss das DOM im Viewtemplate mit diesen Parametern aufgerufen werden:

```
<div tal:define=''mydom python:context.getDom(pOrgNr='1234')''>
    \ldots
</div>
```

3.2 Externe Methode oder externes Viewtemplate zur Darstellung verwenden

Wird das remoteXMLObject hauptsaächlich programmatisch als ProxyObjekt eingesetzt, können beliebige Methoden innerhalb des Acquisitionspfads für die Darstellung und Ausgabe des XML-DOMs verwendet werden. Da hier im Gegensatz zum internen View kein Binding für das DOM zur Verfügung steht, muss das DOM explizit via API (getDOM) abgerufen werden. Danach kann vollkommen analog zur Vorgehensweise bei internen Templates gearbeiet werden. Neben der Python-API ist auch hier für das mit 'getDOM()' geholte Objekt eine XPath-Traversierung via TALES Pfadexpressions möglich.

```
Nachname,Vorname,Titel
<metal:block
```

```
    tal:define="void python:request.RESPONSE.setHeader('Content-Type','text/csv');
    void python:request.RESPONSE.setHeader('Content-Disposition','attachment;;
    filename=hurz.csv')"
    tal:repeat="person python:dom.getElements('//member')">
    <span tal:replace="person/surname"/>,
    <span tal:replace="person/givenName"/>,
    <span tal:replace="person/title |string:"/>
</metal:block>
```

3.3 XSLT zur Darstellung verwenden

Theoretisch kann auch ein XSLT für die Darstellung des XML verwendet werden. Der Einsatz ist jedoch nur in speziellen Ausnahmefällen sinnvoll, da hier der gesamte Portalkontext sowie die Einbindung in das Skinningsystem (Decoration etc.) fehlt. Sinnvoll sind jedoch ggf. Teiltransformationen und Kombination mit einem internen View-Template.

Ein entsprechendes XSLT-File kann durch die Objekt-Aktion 'XSLT' eingegeben bzw. hochgeladen und editiert werden.

4 RemoteXMLObjects für HTML-Seiten anwenden

Prinzipiell können RemoteXMLObjects für alle von XML abgeleiteten Strukturen verwendet werden, also auch für HTML. Dadurch können z.B. via XPath bestimmte Element einer entfernten HTML-Seite extrahiert werden. Diese Fragmente der Seite können als HTML-Strukturen dann in den View des RemoteXMLObjects eingebaut werden. So kann aus der entsprechenden Seite des Studentenwerks München für die Mensa Arcisstrasse der Speiseplan extrahiert werden:

Als Target-URL des Speiseplans für die Arcisstrasse dient:

```
http://www.studentenwerk-muenchen.de/mensa/speiseplan/speiseplan_421_-de.html
```

In dieser Seite werden jeweils als Tabelle im HTML Code die Zusatzsstoffe und das aktuelle Speiseangebot dargestellt. Die beiden Tabellen können durch den XPath-Ausdruck '//table' selektiert werden.

Da der Aufbau der Tabelle aus einzelnen XML/HTML-Elementen jedoch zu aufwendig ist, wird hier nicht mit Elementen oder Elementwerten gearbeitet, sondern mir ganzen Fragmenten des Source-Codes. Dazu dient die Methode getSourceStructures die auf dem DOM oder Subobjekten angewendet werden kann.

```
dom.getSourceStructures('//tables')
```

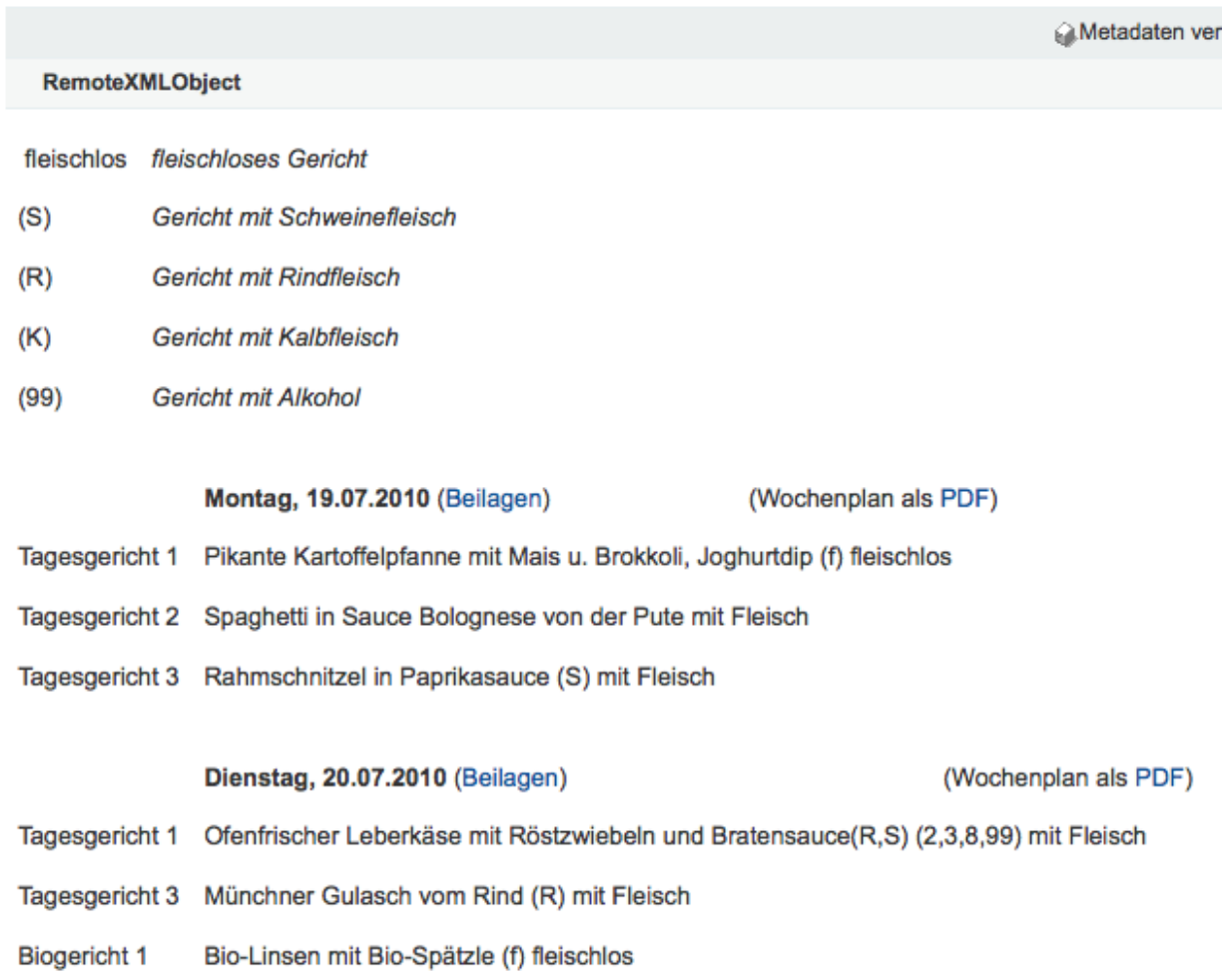
Die Methode liefert zwei Textfragmente mit den Tabellen. Damit können die beiden Tabellen als einzelne Elemente dargestellt werden:. Ein möglicher View sieht dann wie folgt aus:

```
<span tal:repeat="table python:dom.getSourceStructures('//table')"
   tal:replace="structure table">
   Inhalt
</span>
```

Bei Aufruf des RemoteXMLs wird dann der Inhalt entsprechend der Stylesheets des Portals dargestellt. Allerdings werden relative Links auf das Portal bezogen. Um dies zu ändern und sicherzustellen, dass auch relative URLs auf die Orginalseiten zeigen, muss im View noch die Base-URL gesetzt werden:

```
<base href="" tal:attributes="href context/getBaseUrl"/>
<span tal:repeat="table python:dom.getSourceStructures('//table')"
   tal:replace="structure table">
   Inhalt
</span>
```

Nun werden auch die relativen URLs richtig angegeben. Schließlich kann auch noch ein entsprechendes eigenes oder das originale Stylesheet eingebunden werden.



Metadaten ver

RemoteXMLObject

fleischlos	*fleischloses Gericht*
(S)	*Gericht mit Schweinefleisch*
(R)	*Gericht mit Rindfleisch*
(K)	*Gericht mit Kalbfleisch*
(99)	*Gericht mit Alkohol*

	Montag, 19.07.2010 (Beilagen)	(Wochenplan als PDF)
Tagesgericht 1	Pikante Kartoffelpfanne mit Mais u. Brokkoli, Joghurtdip (f) fleischlos	
Tagesgericht 2	Spaghetti in Sauce Bolognese von der Pute mit Fleisch	
Tagesgericht 3	Rahmschnitzel in Paprikasauce (S) mit Fleisch	
	Dienstag, 20.07.2010 (Beilagen)	(Wochenplan als PDF)
Tagesgericht 1	Ofenfrischer Leberkäse mit Röstzwiebeln und Bratensauce(R,S) (2,3,8,99) mit Fleisch	
Tagesgericht 3	Münchner Gulasch vom Rind (R) mit Fleisch	
Biogericht 1	Bio-Linsen mit Bio-Spätzle (f) fleischlos	

Abbildung 12: HTML-Fragment aus Stundenplan der Mensa

5 Python API

5.1 Besonderheiten zur Verwendung von Xpath bei RemoteXMLObjects

Die Adressierung der Elemente des XML-DOM erfolgt über XPath-Ausdrücke. Diese können sowohl in Python-Methoden, sowie in TALES Python-Expressions oder auch TALES-Pathexpressions verwendet werden, wobei bei Path-Expressions eine Traversierung über der XPath simuliert wird.

Während Python alle XPath-Ausdrücke uneingeschränkt unterstützt, ergibt sich bei der Simulation der Traversierung in TALES-Pathexpressions das Problem, dass die XPath-Notation in einigen Fällen (explizit Notationen mit '' und '::' sowie XPath Operators) mit der TALES-Syntax kollidieren. Um dennoch eine weitgehende XPath-Funktionalität via Path-Expressions möglich zu machen, werden hier modifizierte XPath-Ausdrücke verwendet, wobei der '' bzw. '' für Root-Node durch '##' bzw. '#' und der Operator '::' durch '++' ersetzt werden. In relativen Xpath-Ausdrücken (z.B. 'course/courseName') kann der '/' auch in Pathexpressions eingesetzt werden. Aus Kompatibilitätsgründen kann diese modifizierte XPath-Notation auch bei XPath-Audrücken der Python-Methoden verwendet werden. Nicht unterstützt werden zur Zeit XPath Operatoren, wie 'date <= 20.10.2010' etc, da hier eine sinnvolle Ersatzsyntax fehlt und ein Escapen der Ausdrücke problematisch (aber möglich) ist. Die Verwendung in Python-Ausdrücken ist nicht eingeschränkt.

Bei der Verwendung von Elementen mit Namespace kann im TALES-Pfadausdruck die Standardnotation mit '<ns>:<elementname>', z.B. 'tum:news' nicht verwendet werden, da TALES den ':' als Trenner für Datenformate verwendet. Daher muss hier auf die Extended XPath-Definition zurückgegriffen werden. Hier wird der Namespace in geschweiften Klammern dargestellt. 'tum:news' wird daher als '{tum}news' notiert.

Grundsätzlich ist beim Einsatz von modifiziertem XPath in TALES Pathexpressions zu beachten, dass diese Ausdrücke im Gegensatz zur Objekt-Traversieung nicht immer hierarchisch sind (z.B. liefert ##course alle Elemente mit Tag 'course' unabhängig von deren Ebene im DOM) und eben auch mehrdeutig sein können.

5.2 Grundlegendes Verhalten der Methoden

Generell ist das Ergebnis eines XPath-Ausdruck im Gegensatz zu einer URL nie eindeutig, da ein XML-Element mehrere Tochterelemente mit gleichem Tag enthalten kann. Daher liefern alle mit XPath arbeitenden Methoden zunächst grundsätzlich eine Liste von Elementen oder Werten zurück. Mit einer Option kann jeweils nur das erste gefundene Element bzw. der erste gefundene Wert zurückgegeben werden. Bei Resultaten mit nur einem Element oder Wert kann aus Gründen der Handhabung angegeben werden,

dass Anstelle einer Liste direkt der Einzelwert zurückgegeben wird.

5.3 API RemoteXMLObject

Neben den üblichen Attributen und Methoden für die Bearbeitung, Konfiguration und Handhabung des RemoteXMLObject Contentobjekts bietet das RemoteXMLObject einige Methoden die auch für das Rendern des DOM relevant sind. Diese Methoden betreffen das Erstellen des DOM-Objektbaums, das Updaten der XML-Daten vom Remote-Server sowie eine Convenience-Methode um Elemente entsprechend der Werte bestimmter Tochterelemente zu sortieren.

5.3.1 updateData(force=False, **kwargs)

Mit dieser Methode wird das XML für die jeweiligen Sprachen vom Remote-Server mit den entsprechenden Parametern abgerufen. Eventuell zu übergebende Parameter können als Keyword-Arguments übergeben werden. Kommt das entsprechende Argument in der Parameter-Definition des RemoteXMLObject vor, so wird der Argumentwert via Querystring an den Remoteserver übergeben.

Beispiel:

```
<span tal:define="data python:here.updateData(force)">
```

Ein Update des XML wird durchgeführt, wenn der Methode Argumente übergeben werden, die in der Parameterdefinition vorkommen, wenn die eingestellte Updatezeit überschritten ist oder das Flag 'force' gesetzt wurde. Es wird jedoch kein neues DOM erstellt.

5.3.2 getDOM(language=None, **kwargs)

Durch diese Methode wird das XML geparsed und als DOM Element Struktur zurückgegeben, die dann via XPath bearbeitet werden kann.

Wird im Parameter 'language' eine vom Portal unterstützte Sprache übergeben (stelliger ISO-Ländercode), so wird explizit das DOM der entsprechenden Sprache zurückgeliefert. Wird keine Sprache übergeben (Regelfall), so wird die Spracheinstellung des aufrufenden Benutzers verwendet und, wenn die gewünschte Sprache nicht unterstützt wird, die Defaultsprache (Deutsch) verwendet.

Darüberhinaus können weitere Argumente als Keyword-Arguments übergeben werden. Kommt das entsprechende Argument in der Parameterliste des RemoteXMLObjects vor, wird das XML unter Angabe der entsprechenden Parameter neu vom Remote-Server geladen und das DOM neu erstellt.

In der Regel ist es nur notwendig, diese explizite Methode aufzurufen, wenn ein externes View-Template eingesetzt wird oder Parameter übergeben werden sollen, die nicht im Request vorkommen.

Wird der interne View benutzt, so steht das aktuelle DOM in der jeweiligen Sprache durch das Binding 'dom' global zur Verfügung. Ein expliziter Aufruf der Methode 'getDOM' ist nicht notwendig.

5.3.3 sortElements(elementlist, xpath, add_xpath=None, sortorder='asc')

Diese Convenience-Methode erlaubt es, eine als 'elementlist' übergebene Liste von DOM Elementen entsprechend des Wertes des mit 'xpath_sortelement' definierten Tochterelements zu sortieren. Als Sort-Order sind 'asc' für aufsteigende und 'desc' für absteigende Sortierung erlaubt. Der Rückgabewert ist die entsprechend sortierte Liste der Elemente.

5.3.4 singleElementlistelementlist, xpath

Damit können Sie eine Liste von XML-Elementen vereinzeln, wobei ein Element mit der gleichen XPath-Expression nur einmal aufgelistet wird. Diese Methode ist nur vorsichtig einzusetzen, da bei entsprechend grober XPath-Angabe zwei Elemente als gleich angesehen werden weil unter dem angegebenen XPath der gleiche Wert gefunden wird. In diesem Fall würde nur das erste Element in der Liste übernommen, alle folgenden würden ignoriert.

5.3.5 getBaseUrl

Für die Anwendung von RemoteXMLObjects auf exteren HTML-Seiten ist es erforderlich, die BaseURL des remote HTML zu setzen, damit auch relative Links richtig dargestellt werden. Die Methode getBaseUrl ermittelt die Base-URL wenn möglich anhand des '<base>'-Tags oder wenn keines gesetzt ist, wird die entsprechende Target-URL des RemoteXML-Objects verwendet.

5.4 API DOM Element

Das eingelesene XML wird geparsed und als Domain Object Modell (DOM) abgebildet. Die einzelnen Elemente können als Objekt via Python bzw. TALES angesprochen werden. Verschiedene Methoden erlauben das Auslesen von Taginhalt und Attributen die Ausgabe der Tochterelemente, sowie die Query von (Tochter)Elementen via XPath.

5.4.1 getAttributes()

Liefert die Attribute des jeweiligen Elements als Mapping/Dictionary im Format attributname:value zurück.

5.4.2 getAttribute(attributename)

Gibt den Wert des Attributs 'attributename' des aktuellen Elements als Text zurück.

5.4.3 getValue()

Liefert den Textinhalt des aktuellen Elements. Tochterelemente werden ignoriert. Hat das Element selbst keinen Text-Inhalt (z.B. nur Tochterelemente) wird None zurückgeliefert.

5.4.4 getXpath(w3c=False, full=False)

Diese Methode liefert einen (von theoretisch mehreren möglichen) XPath für das aktuelle Objekt bezogen auf das DOM Root-Object an. Der Wert kann für die weitere Traversierung verwendet werden. Ist w3c=True, wird die Zählung der Elemente gleicher Tags mit 0 begonnen, sonst mit 1. Ist das Flag 'full' gesetzt wird nicht der XPfad bezogen auf das DOM angegben, sondern der absolute XPath.

5.4.5 objectValues(tagnames=[])

Diese Methode ist analog zur Methode 'objectValues' von Foldern. Es gibt die Liste aller Tochterelemente in der ersten Ebene zurück deren Tag einem der Werte in 'tagnames' entspricht. Werden keine Tagnames angegeben, so werden alle Tochterelemente zurückgegeben

5.4.6 objectIds(tagnames=[], as_xpath=False)

Analog zur Methode objectValues liefert diese Methode alle Namen der Tags der Tochterelemente in der ersten Ebene des aktuellen Elements zurück, deren Tagname einem der Werte in 'tagnames' entspricht. Wird kein 'tagnames' übergeben, werden die IDs aller Tochterelemente angegeben. Die Reihenfolge der zurückgegebenen IDs entspricht der Reihenfolge der Tags im XML.

5.4.7 getElements(xpath, always_as_list=True, first_only=False)

Hiermit können Sie ausgehend vom aktuellen Element alle Elemente aufrufen, die der übergebenen XPath-Query entsprechen. Werden keine Elemente gefunden ist das Ergebnis None. Mit dem Flag 'first_only' geben Sie an, dass nur das erste Element zurückgegeben wird. Wird das Flag 'always_as_list' auf False gesetzt, so wird, für den Fall, dass nur genau ein Element gefunden wird, dieses nicht als Liste sondern direkt zurückgegeben.

5.4.8 getElementValues(xpath, fallback=None, always_as_list=False, first_only=False)

Mit dieser Methode können die (Text)Werte der unter 'xpath' gefundenen Elemente ausgegeben werden. Die Reihenfolge der Werte entspricht der Reihenfolge der korrespondierenden Elemente. Nicht existierende Werte werden als None oder wenn ein 'fallback' angegeben ist, als dessen Wert zurückgegeben. Beim Flag 'first_only' wird nur der Wert des ersten gefundenen Elements zurückgegeben. Bei nur einem gefundenen Element wird standardmässig immer der Wert des Tags zurückgegeben. Bei gesetztem Flag 'always_as_list' wird auch der Einzelwert als Liste geliefert.

5.4.9 getElementValueList(xpath, children=[])

Durch diese Methode können mehrere Werte von direkten Tochterelementen des Elements bzw. der Elemente unter xpath als Liste ausgelesen werden. Entspricht xpath mehr als ein Element, so werden die Werte für jedes dieser Elemente geliefert.

5.4.10 getElementAttributes(xpath, first_only=True)

Diese Methode liefert eine Liste der Attribute der unter Xpath gefundenen Elemente. Die Attribute eines Elements werden als Mapping der Form attribut:value abgebildet. Bei first_only werden nur die Attribute des ersten gefundenen Elements als Mapping zurückgeliefert.

5.4.11 getSourceStructures(xpath, first_only=True)

Diese Methode liefert den, für die unter 'xpath' gefundenen Elemente, entsprechenden Source-Text inklusive aller Subelemente zurück. Diese Methode ist insbesondere für die Extraktion von HTML-Fragmenten interessant.

5.5 Anwendungsbeispiele

Im Folgenden sollen

```
<publication>
    <title>A brief Survey of riparian Vegetation lower Gaub River</title>
    <date>1997-02-20</date>
    <authors>
        <person>
            <last>Seely</lastname>
            <first>Mary</surename>
        </person>
        <person>
            <last>Jacobsen</lastname>
            <first>Kathy</surename>
        </person>
        <person>
            <last>Wagner</lastname>
            <first>Thomas</surename>
        </person>
    </authors>
</publication>
```

RemoteXMLObject erlaubt einen aus syntaktischen Gründen über TALES Pathexpression nur eingeschränkten Zugriff auf das DOM, da die 'Traversierung' im DOM via XPath erfolgt und die XPath-Expressions teilweise mit Pathexpressions bzw. URI-Notation kollidieren.

Grundsätzlich sind Pathexpressions auf allen Elementen des DOM möglich. Im Gegensatz zu normalen Pfadausdrücken (z.B. heretitle) wird der Pfad auf einem RemoteXML-Element nicht als Objekt- bzw. Attributpfad interpretiert, sondern als XPath-Ausdruck.

Der Rückgabewert ist dabei immer ein Element oder eine Liste von Elementen. Soll der im XML-Tag des Elements enthaltene Text dargestellt werden muss auf dem RemoteXMLElement das Attribut 'text' aufgerufen werden:

Beispiel:

```
<name>
    Mein Name
</name>
```

Hier liefert

```
dom/name
```

das Element mit Tag 'name'. Der eigentliche Wert, also 'mein Name' wird mit

```
dom/name/text
```

dargestellt.

Bei mehreren Elementen mit dem Tag 'Name' liefert

```
dom/name
```

eine Liste aller Elemente mit diesem Tag zurück. Über diese Liste kann dann wie gewohnt iteriert werden. Der eigentliche Text wird dann durch <element>/text ausgegeben.

Beispiele:

```
<span tal:repeat=``name dom/name''>
    <span tal:content=``name/text''/>
</span>

dom/contacts/person[2]/name/given/text

dom/contacts/person[last()]/name/given/text
dom/contacts/person[last()-1]/name/given/text
. . .
```

Problematisch ist die Verwendung von Pfadausdrücken, wenn ein Element ein Tochterelement mit dem Tag 'text' verwendet.

```
<name>
    <text>
        Dies ist mein Name
    </text>
</name>
```

Hier kollidiert die XPath-Expression 'name/text' mit dem TALES Pfadausdruck 'name/text', der nicht das Element 'text' anspricht, sondern das Attribut 'text' des Elements 'name'. Damit ist das Element 'text' über den Tales-Ausdruck nicht anzusprechen. Dies betrifft u.a. Elemente mit den Tags 'attrib', 'text', ... Hier muss über Python-Expressions gearbeitet werden.

Ebenfalls nicht möglich sind Pfadausdrücke bei denen die XPath-Expression mit der TALES-Pfad-Syntax kollidiert, z.B. '//tag', oder '/' tag, da der TALES-Interpreter die '/' nicht als zur XPathe-Expression gehörig erkennen kann. Auch in diesem Fall kann aber über Python-Expressions gearbeitet werden

dom.getElements('//person') statt dem nicht möglichen 'dom///person'

Alternativ kann auch mit einem modifizierten XPath-Ausdruck gearbeitet werden, wobei das '//' durch ein ##' ersetzt wird. Im obigen Beispiel wäre die entspechende TALES-Pathexpression für dom.getElements('//person') der Ausdruck 'dom/##person'

Der Zugriff auf Elemente mit Namespace erfolgt bei TALES-Pfadexpressions nach der Extended XPath-Definition, indem dem eigentlichen Elementname der Namespace in geschweiften Klammern vorangestellt wird:

```
<tum:news>
    <tum:title></tum:title>
</tum:news>
```

Die Adressierung des Elements 'title' wird in TALES Path-Expressions wie folgt abgebildet:

```
dom/{tum}news/{tum}title
```

Dies enstpricht einem dom.getElements('/tum:news/tum:title')

6 Quickreferenz XPath

Im folgenden finden Sie eine Kurzreferenz zu XPath und seiner Anwendung. Weitere Informationen und Details zur XPath-Spezifikation finden Sie auf den Seiten der W3C, http://www.w3.org/TR/xpath20/

6.1 Allgemeines

Ein XML DOM stellt einen Baum aus verschiedenen Elementen dar, die durch ein Tag repräsentiert werden. Das oberste Element (bei RemoteXMLObject das Binding 'dom') wird auch als root bezeichnet. Ein Element kann beliebige Unterelemente haben, dabei sind ein oder mehrere Unterelemente des gleichen Tags möglich.

Im obigen Beispiel ist 'publication' das root-Element mit den Unterelementen 'title', 'date', 'authors'. Das element 'authors' hat drei Unterelemente des Tags 'person'.

Als Parent bezeichnet man das jeweils übergeordnete Element. Im Beispiel ist z.B. 'publication' das Parent von 'title'. Von 'last' mit Wert Wagner ist das Parent das 3. 'person'-Element.

Als Tochterlement bezeichnet man alle Unterlemente der ersten Ebene eines Elements. Also hat das Element 'publication' die Tochterelemente (Children) 'title', 'date', 'authors'. Als Siblings werden alle Elemente bezeichnet, die den selben Parent haben.

Als Descendents bezeichnet an alle Unterelemente eines Elements unabhängig von deren Ebene, also 'authors', 'person', 'last', 'first' sind Descendents von 'publication'. Umgekehrt versteht man unter Ancestors alle Parents eines expliziten Elements bis zum Root.

6.2 Syntax

Xpath verwendet für die Anprache einzelner Elemente überwiegend 'Path Expressions' (analog zu TALES Path Expressions) wobei absolute und relative Pfade möglich sind. Die Adressierung der Elemente erfolgt dabei über den Namen des Tags bzw. da mehrere Tags gleichen Namens als Tochter möglich sind, durch Namen und einen Index

```
tagname             liefert alle Unterelemente des aktuellen Elements mit Tag
                    von Typ 'tagname'
'.'                 liefert das aktuelles Element
'..'                liefert Parent des aktuellen Elements
'/'+tagname         liefert alle Unterelemente vom Typ 'tagname' bezogen auf das
                    root-Element
'//'+tagname        liefert  alle Unterelemente unabhängig von deren Ebene
                    bezogen auf das aktuelle ELement
 '@'                liefert alle Attribute
```

6.3 Predicates

Mit Predicates werden bestimmte, explizite Elemente (z.B. 2. Person-Element im Beispiel) angesprochen oder Elemente die einen expliziten Wert haben.

```
//person[2]
```

Ausserdem können Elemente mit einem bestimmten Wert für ein Tochterelement selektiert werden

```
 //person[last=Wagner]
```

oder Elemente mit einem bestimmten Attribut, hier z.B. <title lang="english">

```
/title[@lang='english']
```

Ausserdem ist es möglich, '*' als Wildcard in Zusammenhang mit Element und Attributselektoren zu verwenden:

```
 //*
 //title[@*]
```

6.4 Location und Axes

Locations sind relative oder absolute Pfad-Ausdücke für ein oder mehrere (Sub)Elemente:

```
/authors/person
/authors/person[1]/last
person/last
```

Neben Pfadausdrücken können auch sogenannte Axes zum Einsatz kommen. Als Axis wird ein Element-Set bezogen auf das aktuelle Element genannt:

```
self                aktuelles Element
ancestor            liefert alle Vorfahren (parent, parents parent etc.) des
                    aktuellen Elements
ancestor-or-self    Wie ancestors, nur dass das aktuelle Element (self) mit
                    eingeschlossen wird
attribute           selektiert alle Attribute des aktuellen Nodes
child               selektiert alle Children des aktuellen Nodes
descendant          selektiert alle Abkömmlinge (children, grandchildren, etc.)
                    des aktuellen Nodes
```

```
descendant-or-self  selektiert alle Abkömmlinge und den Node selbst
following-sibling   selektiert alle Geschwister nach dem aktuellen Node
parent              selektiert den Parent des aktuellen Nodes
preceding           selektiert alle Elemente vor dem Starttag des
                    aktuellen Nodes
preceding-sibling   liefert alle Geschwister vor dem aktuellen Node
self                aktueller Node
```

6.5 Operators

XPath-Operators werden ebenfalls unterstützt. Wegen der Kollision mit TALSE-Path-Expressions ist die Verwendung von Operatoren jedoch weitgehend auf Python-Ausdrücke beschränkt.

7 Bekannte Probleme mit XPath-Ausdrücken

Bei XMLs, die Elemente mit den IDs 'text', 'id' '. . . ' beinhalten kann der Traversionsmechanismus von TALES nicht eingesetzt werden. Diese Namen stehen für echte Attribute der DOM-Elemente, sodass TALES zunächst diese Attribute ausliest und erst wenn es diese nicht findet, das oder die Tochterelemente mit dieser ID verwendet.

Die Verwendung bestimmter XPath-Ausdrücke ('//', ':') und insbesondere die XPath-Operatoren kollidieren mit der TALES-Syntax. Für die gebräuchlichsten Ausdrücken z.B. '//' stehen modifizierte XPath-Ausdrücke zur Verfügung (s.o.), allerdings empfiehlt es sich bei komplexeren XPath-Ausdrücken oder der Verwendung von Operatoren grundsätzlich auf TALES Python-Expressions zurückzugreifen.

Die Fehlerbehandlung und das Debugging bei 'falschen' XPath-Ausdrücken ist derzeit nur rudimentär implementiert. Analog zu Traversierungsfehlern bei TALES werden hier ggf. 'Attribute-Errors' angezeigt oder bei Fallback einfach eine leere Liste oder None zurückgegeben. Das Verhalten ist dabei analog zu TALES und muss ebenso wie bei nicht vorhandenen Attributen mit 'tal:condition' oder 'expression|fallback' berücksichtigt werden.

8 Fehlerbehandlung

8.1 Fehler beim Abruf des XML

Für die Behandlung von Fehlern beim Abruf des XML vom Remote-Server kann der Administrator weitgehende Einstellungen vornehmen. Bei der Konfiguration des RemoteXML kann festgelegt werden, wie sich das RemoteXMLObject im Fehlerfall verhalten soll, d.h. ob sofort eine Fehleranzeige erfolgen soll oder ob zunächst mit gecachten Daten weitergearbeitet werden soll. Dabei kann angegeben werden, ob der Benutzer darüber informiert werden soll, dass die Daten aus dem Cache verwendet werden:

Abbildung 13: Hinweis für Benutzer

Ausserdem kann festgelegt werden, nach welcher Zeit anstelle der gecachten Daten auf die Fehlerseite gesprungen werden soll, wenn der Server bis dahin nicht wieder erreichbar ist.

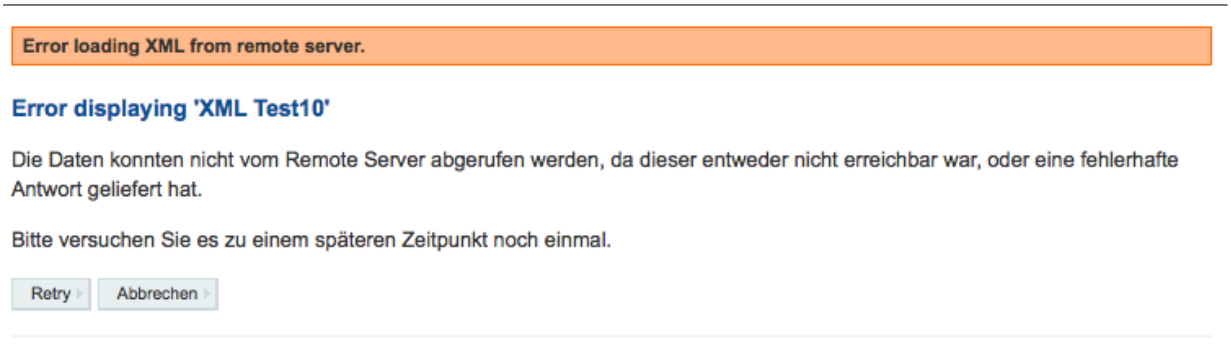


Abbildung 14: Fehlerseite

Schließlich kann Anstelle der Standard-Fehlerseite ein beliebiges Errortemplate angegeben werden, sodass eine individuelle Fehlerbehandlung möglich ist.

8.2 Fehler im View-Template

Prinzipiell gelten für die Erstellung des View-Templates dieselben Bedingungen wie für ZPT-Dokumente, d.h. es werden die üblichen TALES-Fehler ('Attribute Erroro' etc.) gemeldet. Um im Fehlerfall die Bearbeitung des Views zu erleichtern, werden bei RemoteXMl die Fehler jedoch nicht als explizite Fehlerseite dargestellt, sondern in der Bearbeitenansicht des RemoteXML mit einem entsprechenden Hinweis in der Statuszeile eingeblendet:

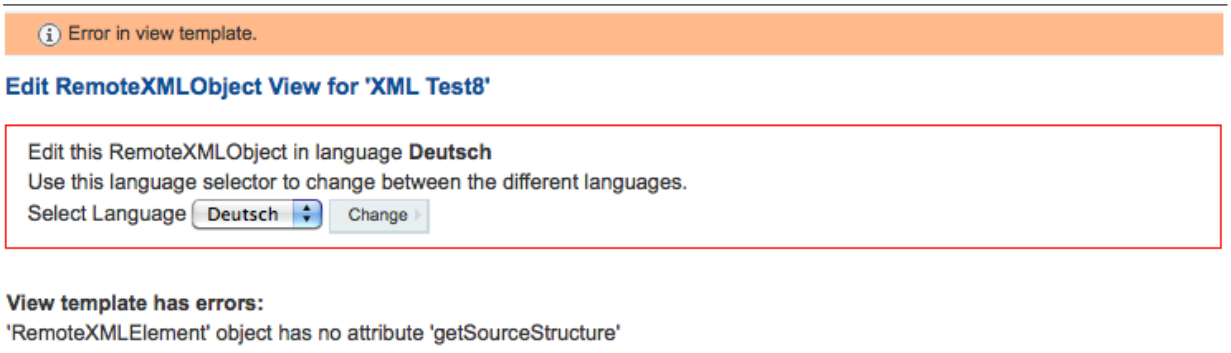


Abbildung 15: Hinweis bei Fehler im View-Template

9 Spezielle Hilfsmethoden für TUMOnline

Für die Einbindung verschiedener Services aus TUMOnline werden Benutzerkennungen und Organisationskennungen benötigt, z.B. um die Mitarbeiter einer Einrichtung abzurufen oder die Vorlesungen einer Person abzufragen.

Da TUMOnline mit anderen Benutzerkennungen arbeitet, sind Methoden erforderlich, die es erlauben für einen Portalbenutzer die zugehörigen TUMOnline-Benutzer-ID bzw seinen Organisationsschlüssel zu ermitteln.

Hierfür existieren eine Reihe von Hilfsmethoden, die im jeweiligen Context aufgerufen werden können.

9.1 getTUMOnlineIds(affiliation=None, mwnid=None, all=False)

Damit können Sie zu einem Portalnutzer seine entsprechende obfuscated TUMOnline-ID ermitteln. Da ein Benutzer in TUMOnline verschiedene IDs haben kann, abhängig von seinem Status z.B. eine als Student, eine als Mitarbeiter und eine als Alumni etc. muss angegeben werden, für welche Affiliation ('Mitarbeiter', 'Student', 'Sonstige') die Obfuscated-ID ermittelt werden soll. Sollen alle IDs ermittelt werden ist das Flag 'all' auf True zu setzen. Wird eine Affiliation übergeben, so erhalten Sie nur die entsprechende ID. Wird keine MWNID übergeben, so wird immer der aufrufende Benutzer angenommen, bei übergebener MWNID wird die ID für den entsprechenden Benutzer ermittelt.

9.2 getMWNIDbyObfuscatedId(obfuscatedid)

Diese Methode ermittelt bei einer gegebenen obfuscated TUMOnline-ID die MWNID des entsprechenden Benutzers im myTUM-Portal. Wird kein passender Benutzer gefunden, so wird None zurückgegeben.

```
<span tal:replace=\"python:here.getMWNIDbyObfuscatedId('C7260010E7D20843')"/>
```

9.3 Ermittlung der Organisationszugehörigkeit einer Person

Um die Organisationszugehörigkeit einer Person zu ermitteln, muss das Property 'tumUnivisKey' des jeweiligen Benutzers abgefragt werden. Die Bezeichnung hat historische Gründe, das Property hat jedoch die gültigen TU-Org-Kennungen gespeichert.

```
<span tal:replace=\"python:member.getProperty('tumUnivisKey',None)/>
```