XML magazin & WEB SERVICES

XML magazin & WEB SERVICES

Deutschland € 9,80
Österreich € 10,20
Luxemburg € 11,25
Schweiz SFr 19,20

S&S Technology | Integration | Business

1.04

Business Processes

Management, Visualisierung, Optimierung

Sichere Web Services

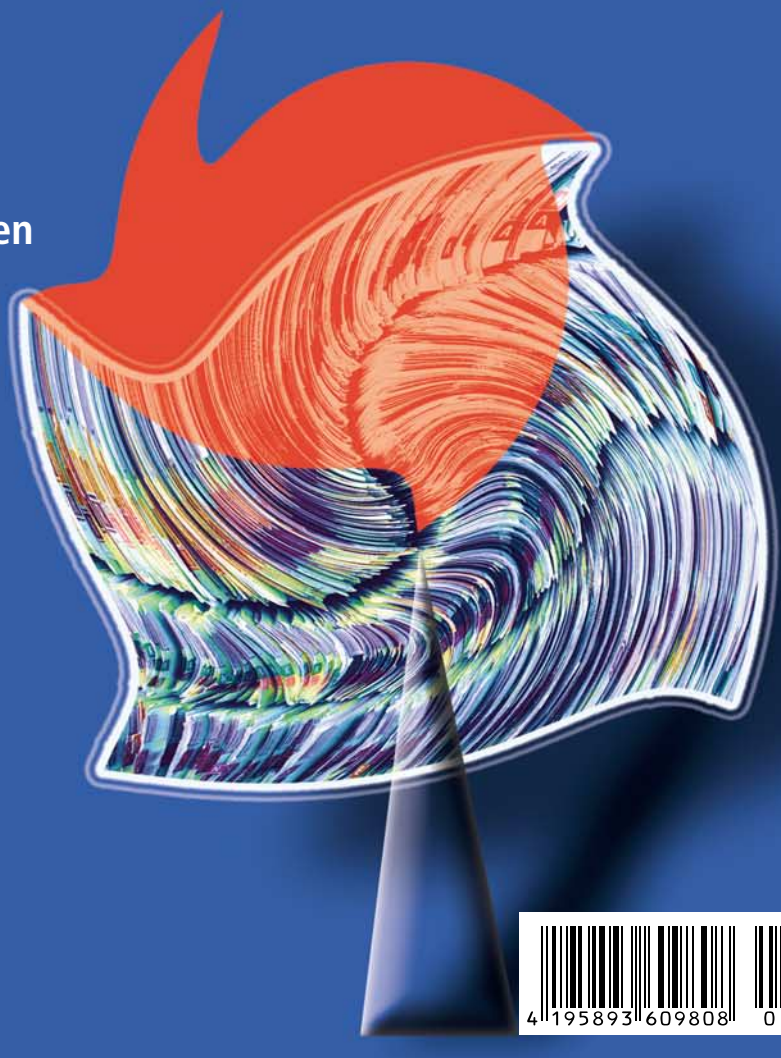
WS-Security und SAML

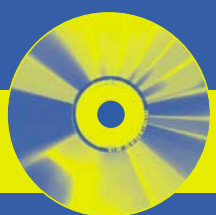
XML ressourcenschonend
Kompaktere XML-Dokumente erstellen

PHP und Web Services
Die eigene Weblog-Applikation

XSLT-Praxis
Skripte für OpenOffice XML Filter

Geräteunabhängige Webformulare
Das Consensus-Projekt





mit CD!

4 195893 609808 01

Chemiestunde

Entwicklung von dynamischen Datenzugriffstechniken für Scalable Vector Graphics

XML und Chemie haben durchaus Berührungspunkte. Da wäre zunächst die Chemical Markup Language (CML) zur Beschreibung chemischer Strukturen zu nennen [1], die jedoch nicht Gegenstand dieses Artikels ist. Außerdem bietet sich XML auch für Chemiker zur Ablage und Verteilung von Daten an. Hier soll ein Web Service vorgestellt werden, der Informationen zu den chemischen Elementen liefert. Im Mittelpunkt steht der Zugriff auf den Dienst aus einem SVG-Dokument heraus.

von Thomas Meinike

Rückblick

In [2] wurde gezeigt, wie sich SVG-Dokumente mit externen Datenquellen verbinden lassen. Die behandelten Methoden *getURL()*, *postURL()* sowie *parseXML()* stehen bis heute nur bei der Verwendung des Adobe SVG Viewers zur Verfügung [3]. Mittlerweile hat das W3C diese Techniken als Erweiterungen für die im Stadium eines Working Drafts befindliche Spezifikation SVG 1.2 vorgesehen [4]. Somit kann man davon ausgehen, dass künftig auch andere Anbieter von Plug-ins entsprechende Schnittstellen integrieren.

Periodic Table Web Service

Die Idee, einen Web Service mit einem SVG-Dokument zu „verheiraten", entstand beim Stöbern durch die Angebote von WebserviceX.NET. Neben Wetterdaten, Währungsumrechnungen und weiteren interessanten Diensten lassen sich auch Daten zu den chemischen Elementen abrufen. Dieser Periodic Table genannte Web Service [5] wird über den Aufruf von *http://www.webservicex.net/periodictable.asmx* gesteuert. Neben der WSDL-Beschreibung (Parameter *?WSDL*) können durch die folgenden Anhänge vier weitere *GET*-Abfragen realisiert werden:

- */GetAtoms* (Auflistung der chemischen Elementnamen)
- */GetAtomicWeight?ElementName=...* (Atomgewicht eines Elements)
- */GetAtomicNumber?ElementName=...* (u. a. Ordnungszahl eines Elements)
- */GetElementSymbol?ElementName=...* (Symbol eines Elements)

Die Punkte nach dem Gleichheitszeichen sind jeweils durch einen konkreten Elementnamen zu ersetzen, die möglichen Werte von *A*(ctinium) bis *Z*(irconium) entnimmt man der von *GetAtoms* bereitgestellten Liste. Der Web Service liefert zunächst diese XML-Datenhülle:

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://www.webserviceX.NET">

  ... eigentlicher Inhalt, < und > als Entitys kodiert ...

</string>
```

Fragt man mittels *http://.../periodictable.asmx/GetElementSymbol?ElementName=Iridium* nach dem Symbol des Elements Iridium, erhält man:

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://www.webserviceX.NET">&lt;
                                    NewDataSet&gt;
 &lt;Table&gt;
  &lt;Symbol&gt;Ir&lt;/Symbol&gt;
 &lt;/Table&gt;
&lt;/NewDataSet&gt;</string>
```

Man kann hier bereits erahnen, dass die Teilzeichenkette *Ir* der gesuchten Information entspricht. Eine weitere Verarbeitung der innerhalb des Elements *string* befindlichen Ergebnisdaten ist also erforderlich. Das Atomgewicht lässt sich analog ermitteln und schließlich bleibt noch *GetAtomicNumber* übrig. Entgegen der Erwartung eines einzelnen Wertes erhält man einen ganzen Datensatz in dieser Reihenfolge:

- Ordnungszahl
- Elementname
- Elementsymbol
- Atomgewicht (in atomaren Einheiten)
- Siedetemperatur (in Kelvin)
- Ionisationspotenzial (in Elektronenvolt)
- Elektronegativität (Zahlenwert ohne Einheit)
- Atomradius (in Ångström, 1 Å = 10^{-10} m)
- Schmelztemperatur (in Kelvin)
- Dichte (in Kilogramm pro Kubikmeter)

Abbildung 1 zeigt die Ergebnisstruktur für Iridium beim direkten Aufruf im Internet Explorer. Die verwendeten XML-Elementnamen lauten *AtomicNumber*, *ElementName*, *Symbol*, *AtomicWeight*, *BoilingPoint*, *IonisationPoten-*

Quellcode

Den Quellcode zum Artikel finden Sie auf der beiliegenden CD.

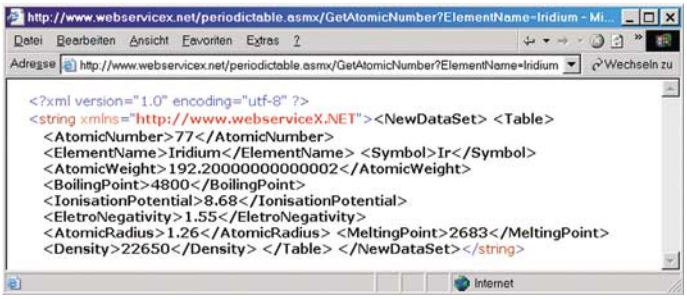


Abb. 1: Von *GetAtomicNumber* erzeugter XML-Datensatz für das Element Iridium.

tial, *EletroNegativity* (hier fehlt offenbar ein „c“), *AtomicRadius*, *MeltingPoint* und *Density*. Liegen bestimmte Daten zu einem chemischen Element nicht vor, erscheinen auch die zugehörigen XML-Elemente nicht, sodass man beim Zugriff auf die Ergebnisse entsprechend reagieren muss.

SVG ruft Web Service

Der beschriebene Web Service legte nahe, eine SVG-Repräsentation des Periodensystems der Elemente aufzubauen und mit den Online-Daten zu verbinden (*periodensystem.svg*). Beim Anklicken der Kästchen mit den Elementsymbolen erfolgt zunächst über den Event Handler *onclick* der Aufruf der JavaScript-Funktion *HoleDaten()*, die den Namen des aktuellen Elements als Argument erhält:

```
function HoleDaten(EName)
{
 // Beispielaufruf von ptable.php:
 // http://.../ptable.php?EName=Actinium
 getURL("ptable.php?EName="+EName,callback);
}
```

Über *getURL()* wird der Elementname an das PHP-Skript *ptable.php* weitergereicht, welches den Web Service anspricht und die Rohdaten entgegennimmt sowie einige Konvertierungen ausführt (Entities in spitze Klammern umwandeln, überflüssige Zeichenketten abtrennen). Tritt ein Fehler auf, wird *Error!* geliefert (siehe Listing 1).

Listing 1

```
<?php

//-------------------------------------------------------------------
// ptable.php holt Periodic Table-Daten von WebserviceX.NET
// http://www.webservicex.net/periodictable.asmx
//-------------------------------------------------------------------
// by Dr. Thomas Meinike 09/03
// Beispielaufruf: http://.../ptable.php?EName=Actinium
//-------------------------------------------------------------------

function Error()
{
 header("Content-type: text/plain");
 print "Error!"; // Verarbeitung im SVG-Dokument
 exit();
}

// PHP-Version ermitteln ("4.3.3" wird zu 433)
$verstr=explode(".",phpversion());
$vernum=$verstr[0]*100+$verstr[1]*10+$verstr[2]*1;

// Parameter Elementname auslesen
$EName=$_GET["EName"];
if(!isset($EName))Error();

// URL zusammen setzen
$url="http://www.webservicex.net/periodictable.asmx/GetAtomicNumber?
                                                    ElementName=$EName";

// Variablen fuer Ergebnisdaten
$ausgabe="";
$xmlcontent="";

// Daten vom Web Service auslesen
if($vernum >= 430)$xmlcontent=file_get_contents($url);
else
{
 $fp=@file($url);
 $xmlcontent=implode("",$fp);
}

// Ergebnisstring aufbereiten
$xmlcontent=trim($xmlcontent);
$xmlcontent=str_replace("<string xmlns=\"http://www.webserviceX.NET\">",
                                                              "",$xmlcontent);
$xmlcontent=str_replace("</string>","",$xmlcontent);

if($vernum >= 430)$xmlcontent=html_entity_decode($xmlcontent);
else
{
 $xmlcontent=str_replace("&lt;","<",$xmlcontent);
 $xmlcontent=str_replace("&gt;",">",$xmlcontent);
}

if(strlen($xmlcontent)>54)
{
 /*
  54 entspricht der leeren Rueckgabe von
  <?xml version="1.0" encoding="utf-8"?>
  <NewDataSet />
 */

 // XML-Code ausgeben
 header("Content-type: application/xml");
 print $xmlcontent;
 // Verarbeitung des XML-Inhalts im SVG-Dokument
 // periodensystem.svg mit der callback()-Funktion
}
else Error();

?>
import javax.xml.transform.TransformerFactory;
```

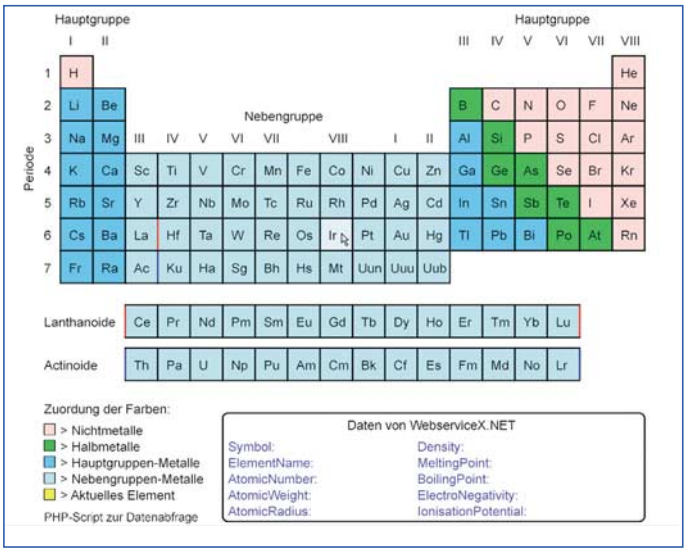


Abb. 2: SVG-Periodensystem der Elemente mit Kopplung an den Periodic Table Web Service

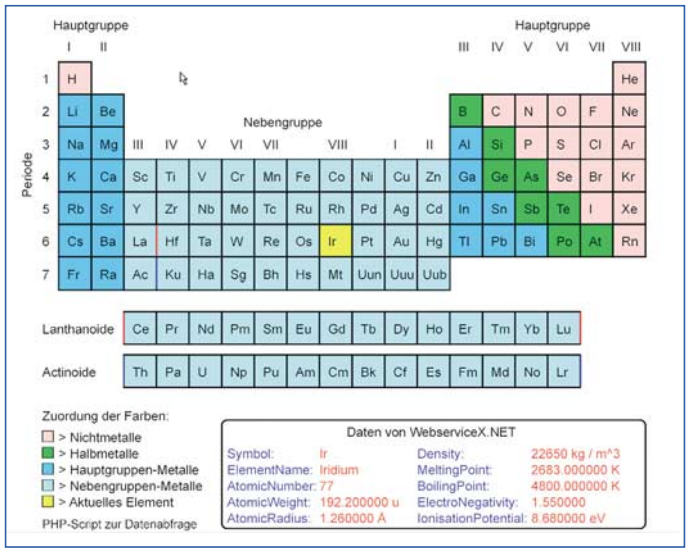


Abb. 3: SVG-Periodensystem der Elemente mit Informationen für das Element Iridium

Listing 2

```
function callback(urlRequestStatus)
{
 if(urlRequestStatus.success && urlRequestStatus.
                                    content!="Error!")
 {
  xmldoc=parseXML(urlRequestStatus.content).
                                    documentElement;
  if(xmldoc)BearbeiteDaten();
 }
 else
 {
  // Fehlermeldung ausgeben
  alert("Bei der Abfrage von WebserviceX.NET\
                                nist ein Fehler aufgetreten!");
 }
}
```

Listing 3

```
function BearbeiteDaten()
{
 // Ausschnitt fuer BoilingPoint

 //XML-Daten auslesen
 BPobj=xmldoc.getElementsByTagName("BoilingPoint");
 if(BPobj && BPobj.length>0)BP=BPobj.item(0).
                                    firstChild.nodeValue;
 else BP="";
 //...

 // einige Werte auf max. 6 Stellen runden und ggf.
                                    Einheiten anfuegen
 if(BP!="")BP=parseFloat(BP).toFixed(6)+" K";
 //...

 // Ergebnisse als Textknoteninhalte ausgeben
 svgdoc.getElementById("BPout").firstChild. nodeValue=BP;
 //...
}
```

Nach dem Abarbeiten des PHP-Codes erfolgt der Aufruf weiterer JavaScript-Funktionen. *callback()* übernimmt den XML-Datensatz unter Verwendung von *parseXML()* (siehe Listing 2). Weitere Details wurden bereits in [2] erläutert.

Die Hilfsfunktion *BearbeiteDaten()* zerlegt mit DOM-Methoden das erhaltene XML-Dokumentfragment und platziert die ermittelten Werte in der Ausgabebox unterhalb des Periodensystems (siehe Listing 3).

Zusätzliche Funktionen haben eher kosmetische Aufgaben: Markieren des aktuellen Elements und Realisierung von Rollover-Effekten durch Veränderung der Opazität der als Rechtecke angelegten Kästchen (CSS-Eigenschaft *opacity*). Abbildungen 2 und 3 zeigen das SVG-Periodensystem nach dem Laden und nach dem Abrufen der Informationen für das Element Iridium.

Fazit

Das verwendete Beispiel sollte das Potenzial der Kommunikation von SVG-Dokumenten mit der „Außenwelt" anschaulich ins Blickfeld rücken. Mit der Etablierung von SVG 1.2 stehen die verwendeten Techniken offiziell zur Verfügung, was auf eine breitere Anwendung hoffen lässt. Der komplette Beispielcode ist auf der Heft-CD enthalten und online unter [6] zu finden. ●

Dr. Thomas Meinike hat Chemie studiert und parallel dazu Erfahrungen im IT-Bereich als Programmierer, Autor und Referent gesammelt. Seit 1997 ist er an der Fachhochschule Merseburg als Lehrkraft mit den Schwerpunkten Online-Dokumentation und Website-Entwicklung tätig. Er ist unter thomas.meinike@et.fh-merseburg.de erreichbar.

Links & Literatur

[1] Details zur CML:
www.xml-cml.org/

[2] T. Meinike, „SVG ruft Datenbank", *XML & Web Services Magazin* 3.2003, S. 66-69.

[3] Adobe:
ASV 3.01 (Sicherheitsupdate von 3.0!):
www.adobe.com/svg/viewer/install/main.html
ASV 6.0 (Preview):
www.adobe.com/svg/viewer/install/beta.html
ASV 3.0 für Linux (Beta):
www.adobe.com/svg/viewer/install/old.html

[4] W3C, SVG 1.2 Working Draft:
www.w3.org/TR/SVG12/#WindowObject

[5] Periodic Table Web Service:
www.webservicex.net/WS/WSDetails.aspx?WSID=19&CATID=7

[6] Beispiel online:
www.datenverdrahten.de/svglbc/?code=periodensystem