H. Neuroth, A. Oßwald, R. Scheffel, S. Strathmann, K. Huth (Hrsg.)

nestor Handbuch

Eine kleine Enzyklopädie der digitalen Langzeitarchivierung

Version 2.3

Kapitel 13.3

Das JSTOR/Harvard Object Validation Environment (JHOVE)

nestor

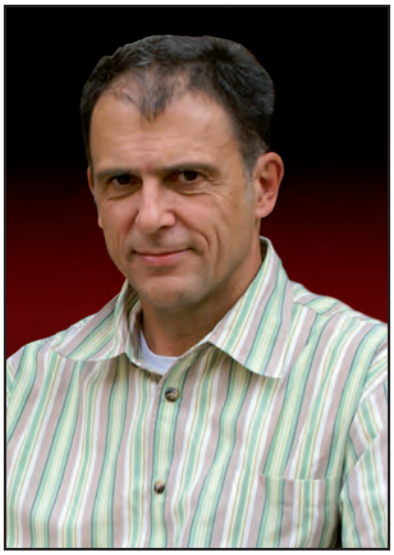
nestor Handbuch: Eine kleine Enzyklopädie der digitalen Langzeitarchivierung
hg. v. H. Neuroth, A. Oßwald, R. Scheffel, S. Strathmann, K. Huth
im Rahmen des Projektes: nestor – Kompetenznetzwerk Langzeitarchivierung und Langzeitverfügbarkeit digitaler Ressourcen für Deutschland
nestor – Network of Expertise in Long-Term Storage of Digital Resources
http://www.langzeitarchivierung.de/

Kontakt: editors@langzeitarchivierung.de
c/o Niedersächsische Staats- und Universitätsbibliothek Göttingen,
Dr. Heike Neuroth, Forschung und Entwicklung, Papendiek 14, 37073 Göttingen

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet unter http://www.d-nb.de/ abrufbar.

Neben der Online Version 2.3 ist eine Printversion 2.0 beim Verlag Werner Hülsbusch, Boizenburg erschienen.

URL für Kapitel 13.3 „Das JSTOR/Harvard Object Validation Environment (JHOVE)“ (Version 2.3): urn:nbn:de:0008-20100617276
http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:0008-20100617276



Gewidmet der Erinnerung an Hans Liegmann (†), der als Mitinitiator und früherer Herausgeber des Handbuchs ganz wesentlich an dessen Entstehung beteiligt war.

13.3 Das JSTOR/Harvard Object Validation Environment[18] (JHOVE)

Stefan E. Funk

Einführung

Wie in den vorangehenden Kapiteln bereits besprochen wurde, ist es für eine langfristige Erhaltung von digitalen Objekten dringend erforderlich, zu wissen und zu dokumentieren, in welchem Dateiformat ein solches digitales Objekt vorliegt. Zu diesem Zweck sind auch Informationen von Nutzen, die über das Wissen über den Typ eines Objekts hinausgehen, vor allem detaillierte technische Informationen. Zu wissen, dass es sich bei einem digitalen Bild um ein TIFF-Dokument in Version 6.0 handelt, reicht evtl. nicht aus für eine sinnvolle Langzeiterhaltung. Hilfreich können später Daten sein wie: Welche Auslösung und Farbtiefe hat das Bild? Ist es komprimiert? Und wenn ja, mit welchem Algorithmus? Solche Informationen – technische Metadaten – können aus den Daten des Objekts selbst (bis zu einem gewissen Grad, welcher vom Format der Datei abhängt) automatisiert extrahiert werden.

Anwendung

Mit JHOVE wird im Folgenden ein Werkzeug beschrieben, das außer einer Charakterisierung einer Datei (Welches Format liegt vor?) und einer Validierung (Handelt es sich um eine valide Datei im Sinne der Format-Spezifikation?) zu guter Letzt auch noch technische Metadaten extrahiert. JHOVE kann entweder mit einem grafischen Frontend genutzt werden – wobei eine Validierung oder Extraktion technischer Metadaten von vielen Dateien nicht möglich ist, oder als Kommandozeilen-Tool. Ebenso kann JHOVE auch direkt als Java-Anwendung in eigene Programme eingebunden werden, was für eine automatisierte Nutzung sinnvoll ist. Letzteres ist jedoch dem erfahrenen Java-Programmierer vorbehalten. Als Einführung wird hier das grafische Frontend kurz erklärt sowie eine Nutzung auf der Kommandozeile beschrieben.

---

18 JHOVE – JSTOR/Harvard Object Validation Environment: http://hul.harvard.edu/jhove/

Anforderungen

Für die Nutzung von JHOVE wird eine Java Virtual Machine benötigt, auf der JHOVE Projektseite bei Sourgeforge.net[19] wird Java in Version 1.6.0_12 empfohlen.

Das grafische Frontend JhoveView

Download

Nach dem Herunterladen des .zip oder .tar.gz Paketes von der Sourceforge Projektseite – beschrieben wird hier die Version 1.2 vom 10. Februar 2009 – wird das Paket in ein beliebiges Verzeichnis entpackt. Zum Starten des grafischen Frontends starten Sie bitte das Programm JhoveView.jar im Verzeichnis ./bin/ – entweder durch Doppelklick oder von der Kommandozeile per java -jar bin/JhoveView.jar (nach dem Wechsel in das Verzeichnis, indem sich JHOVE befindet).

Menü-Optionen

Die beiden vorhandenen Menü-Optionen "File" und "Edit" sind schnell erklärt:

- Unter "File" kann eine Datei aus dem Internet oder vom Dateisystem geöffnet werden, das sogleich von JHOVE untersucht wird.
- Unter "Edit" kann gezielt ein JHOVE-Modul gewählt werden, mit dem eine Datein untersucht werden soll. Nicht die Einstellung "(Any)" zu benutzen – für eine automatische Erkennung des Formats – kann zum Beispiel dann Sinn machen, wenn eine TIFF-Datei nicht automatisch als solche erkannt wird, weil sie vielleicht nicht valide ist. Dann kann JHOVE dazu bewegt werden, dieses Bild mit dem TIFF-Modul zu untersuchen, um so eine entsprechende – und weiter helfende – Fehlermeldung zu bekommen. Weiterhin kann hier die Konfigurationsdatei editiert werden (um neue Module einzubinden).

19 http://sourceforge.net/projects/jhove/

Dateien untersuchen

Wählt man nun eine Datei aus, für erste Tests sollten die vorhandenen Module berücksichtigt werden, wird diese Datei von JHOVE untersucht. Im Folgenden wird ein Fenster angezeigt, in dem alle von JHOVE extrahierten Informationen angezeigt werden. Hier kann nach Belieben durch den Baum geklickt werden. An erster Stelle wird das Modul und dessen Versionsnummer angezeigt, mit dem die Datei untersucht wurde. Wird hier als Modulname "BYTESTREAM" angezeigt, heißt das, dass JHOVE kein passendes Modul gefunden hat, das Bytestream-Modul wird dann als Fallback genutzt. Hier hilft es unter Umständen – wie oben erwähnt – das Modul per Hand einzustellen.

JHOVE Ausgaben anzeigen und speichern

Die Speicheroption, die nun zur Verfügung steht, kann genutzt werden, um die Ergebnisse wahlweise als Text oder als XML zu speichern und in einem anderen Programm zu nutzen/anzusehen. So können die Informationen beispielsweise in einem XML- oder Texteditor bearbeitet oder anderweitig genutzt werden. Im Folgenden ein Beispiel einer Untersuchung einer Textdatei im Zeichensatz UTF-8:

```
JhoveView (Rel. 1.1, 2008-02-21)
       Date: 2009-03-03 10:33:31 CET
       RepresentationInformation:
       /Users/fugu/Desktop/nestor-hand
       buch-kapitel-13_2009-03-03/test.txt
       ReportingModule: UTF8-hul, Rel. 1.3 (2007-08-30)
       LastModified: 2009-03-03 10:33:12 CET
       Size: 64
       Format: UTF-8
       Status: Well-Formed and valid
MIMEtype: text/plain; charset=UTF-8
UTF8Metadata:
       Characters: 60
       UnicodeBlocks: Basic Latin, CJK Unified Ideographs
```

Als XML-Repräsentation sieht das Ergebnis aus wie folgt und kann somit maschinell sehr viel genauer interpretiert werden.

```
<?xml version="1.0" encoding="utf-8"?>
<jhove xmlns:xsi="http://www.w3.org/2001/XMLS-
chema-instance" xmlns="http://hul.harvard.edu/
ois/xml/ns/jhove" xsi:schemaLocation="http://
```

```
      hul.harvard.edu/ois/xml/ns/jhove http://hul.
      harvard.edu/ois/xml/xsd/jhove/1.5/jhove.xsd"
      name="JhoveView" release="1.1" date="2008-02-21">
      <date>2009-03-03T10:40:00+01:00</date>
      <repInfo
      uri="/Users/fugu/Desktop/nestor-hand-
buch-kapitel-13_2009-03-03/test.txt">
      <reportingModule release="1.3"
date="2007-08-30">UTF8-hul</reportingModule>
      <lastModified>2009-03-03T10:33:12+01:00</lastModified>
      <size>64</size>
      <format>UTF-8</format>
      <status>Well-Formed and valid</status>
      <mimeType>text/plain; charset=UTF-8</mimeType>
      <properties>
            <property>
                  <name>UTF8Metadata</name>
                  <values arity="List" type="Property">
                        <property>
                              <name>Characters</name>
                              <val-
ues arity="Scalar" type="Long">
                                    <value>60</value>
                              </values>
                        </property>
                        <property>
                              <name>UnicodeBlocks</name>
                              <val-
ues arity="List" type="String">

<value>Basic Latin</value>

<value>CJK Unified Ideographs</value>
                              </values>
                        </property>
                  </values>
            </property>
      </properties>
      <note>Additional representation in-
formation includes the line endings:
            CR, LF, or CRLF</note>
      </repInfo>
      </jhove>
```

Eine genauere Dokumentation des grafischen Frontends, des Kommandozeilentools, sowie zu JOHVE allgemein findet sich auf der JHOVE-Homepage (auf Englisch) unter "Tutorial", aktuelle Informationen zur Distribution und die neueste Version derselben auf der JHOVE SourceForge-Projektseite.

JHOVE auf der Kommandozeile

Die Möglichkeit, ganze Verzeichnisse zu untersuchen und kurz mal zu schauen, wieviele valide Dateien darin enthalten sind, ist – neben allen Möglichkeiten des grafischen Frontends – ein großer Vorteil des Kommandozeilentools, das JHOVE zur Verfügung stellt.

Konfiguration

Um das Kommandozeilentool nutzen zu können, ändern Sie bitte zunächst den Namen der Datei jhove.tmpl in jhove (Linux/Unix) oder jhove_bat.tmpl in jhove.bat (Windows). Ändern Sie bitte noch – den Anweisungen in diesen Dateien zufolge – den Pfad zu Ihrem JHOVE-Verzeichnis in diesen Skripten. Haben Sie beispielsweise das JHOVE-Paket in /home/ kopiert, lautet der Pfad /home/jhove (Linux/Unix), arbeiten Sie auf einem Windows-System, tragen Sie für das Verzeichnis C:\Programme\ bitte C:\Programme\jhove ein. Sollte der Pfad zu Ihrer Java-Installation nicht stimmen, passen Sie bitte auch diesen noch an. Wenn Sie alles richtig konfiguriert haben, bekommen Sie durch Tippen von ./jhove bzw. jhove.bat detaillierte Informationen zu Ihrer JHOVE-Installation.

Verzeichnisse rekursiv untersuchen

Wenn Sie nun beispielsweise alle XML-Dateien untersuchen möchten, die sich im Beispiel-Verzeichnis der JHOVE-Installation befinden, rufen Sie JHOVE folgendermaßen auf:

```
./jhove -h audit examples/xml/
```

Die Ausgabe enthält folgendes und beschreibt in Kürze, welche Dateien untersucht wurden, ob und wie viele davon valide sind:

```
<?xml version="1.0" encoding="UTF-8"?>
<jhove xmlns:xsi="http://www.w3.org/2001/XMLSche-
ma-instance" xmlns="http://hul.harvard.edu/
ois/xml/ns/jhove" xsi:schemaLocation="http://
hul.harvard.edu/ois/xml/ns/jhove http://hul.
harvard.edu/ois/xml/xsd/jhove/1.5/jhove.xsd"
name="Jhove" release="1.1" date="2008-02-21">
```

```
<date>2009-03-03T11:27:27+01:00</date>
<audit home="/Users/Fugu/Desktop/jhove">
<file mime="text/xml" status="well-formed">
examples/xml/build.xml</file>
<file mime="text/plain; charset=US-ASCII" status="valid">
examples/xml/external-parsed-entity.ent</file>
<file mime="text/plain; charset=US-ASCII" status="valid">
examples/xml/external-unparsed-entity.ent</file>
<file mime="text/xml" status="well-formed">
examples/xml/jhoveconf.xml</file>
<file mime="text/plain; charset=US-ASCII" status="valid">
examples/xml/valid-external.dtd</file>
</audit>
</jhove>
<!-- Summary by MIME type:
text/plain; charset=US-ASCII: 3 (3,0)
text/xml: 2 (0,2)
Total: 5 (3,2)
-->
<!-- Summary by directory:
/Users/Fugu/Desktop/jhove/examples/xml: 5 (3,2) + 0,0
Total: 5 (3,2) + 0,0
-->
<!-- Elapsed time: 0:00:02 >
```

Weitere Parameter

Als weitere Parameter können unter anderem Handler und Module genauer spezifiziert werden sowie Ausgabe-Dateien und Encoding konfiguriert werden. Hier darf nach Belieben probiert, getestet und gespielt werden, um zu probieren, technische Metadaten zu extrahieren und Dateien zu validieren.
Im Folgenden noch eine kurze Beschreibung des Nutzung des Kommandozeilentools...

```
jhove  [-c config] [-m module [-p param]]
       [-h handler [-P param]]
       [-e encoding][-H handler] [-o output] [-x saxclass]
       [-t tempdir] [-b bufsize] [[-krs] dir-file-or-uri […]]
```

...und die Bedeutung der wichtigsten:

`-c config`	–	Pfad zur JHOVE-Konfigurationsdatei.
`-m module`	–	Name des Moduls, möglich sind hier: AIFF-hul, ASCII-hul, BYTESTREAM, GIF-hul, HTML-hul, JPEG-hul, JPEG2000-hul, PDF-hul, TIFF-hul, UTF8-hul, WAVE-hul und XML-hul.
`-p param`	–	Modul-spezifische Parameter.
`-h handler`	–	Name des Output-Handlers (Grundeinstellung: TEXT).
`-P param`	–	Handler-spezifische Parameter.
`-o output`	–	Name der Ausgabe-Datei (Grundeinstellung: stdout).
`-x saxclass`	–	SAX-Parser-Klasse (Grundeinstellung: J2SE 1.4 default).
`-t tempdir`	–	Temporäres Verzeichnis, in dem temporäre Dateien erzeugt werden.
`-b bufsize`	–	Puffergröße für gepufferte I/O Operationen (Grundeinstellung: J2SE 1.4 default).
`-k`	–	Berechnet CRC32, MD5, und SHA-1 Checksummen.
`-r`	–	Zeigt rohe Data Flags an, nicht die textlichen Äquivalente.
`-s`	–	Format-Identifikation basiert nur auf internen Signaturen.

`dir-file-or-uri`	–	Verzeichnis, Pfadname oder URI der zu untersuchenden Dateien.