Dr. Oliver Diedrich

Der kleine E-Autor

E-Books im Epub-Format selbst erstellen

Wenn das Angebot an kaufbaren E-Books zu dünn ist, muss man eben selbst Hand anlegen. Eigene Inhalte und im Internet frei verfügbare Texte lassen sich mit etwas Handarbeit oder vollautomatisiert per Software in E-Books nach dem Epub-Standard verwandeln, die fast jeder Reader anzeigen kann.

Die Auswahl an E-Books auf dem deutschen Markt ist nach wie vor bescheiden: Nicht nur, dass aktuelle Bestseller kaum in elektronischer Form erhältlich sind, auch die allermeisten älteren Bücher sind nicht als E-Book zu kriegen. Dabei kann man beim Projekt Gutenberg oder bei Zeno.org zahlreiche Klassiker von Goethe bis Tucholsky, von Dostojewski bis Ringelnatz online lesen. Möglich ist das, weil das Urheberrecht 70 Jahre nach dem Tod des Autors abläuft. Lesestoff ist also genügend da.

Aber wie kriegt man diese Bücher auf seinen E-Book-Reader? Oder produziert selbst ein E-Book aus eigenen Inhalten? Das ist gar nicht so kompliziert: Mit ein paar Skripten oder etwas Handarbeit lassen sich Texte, die in HTML oder XML ausgezeichnet sind oder in einem anderen strukturierten Format vorliegen, in E-Books im Epub-Format umwandeln. Alles, was dazu nötig ist, sind rudimentäre HTML-Kenntnisse.

Das E-Book-Format Epub ist ein vom International Digital Publishing Forum (IDPF, www.idpf.org) definiertes offenes XML-Format zur Aufbereitung von Inhalten für E-Books, das praktisch alle E-Book-Reader lesen können – mit Ausnahme des Kindle, der auf Amazons proprietäres Format festgelegt ist. Das IDPF, dessen Mitgliederliste fast 150 Firmen von Adobe über Amazon bis zu O'Reilly und Random House umfasst, will das digitale Publizieren durch die Entwicklung offener Standards fördern.

Anders als das PDF-Format, das das Seiten-Layout pixelgenau festlegt und daher für die kleinen Displays der Lesegeräte nur bedingt geeignet ist, werden Inhalte in Epub-Dateien mit HTML (genau gesagt: in dem strengeren, in XML definierten XHTML) oder in dem vom DAISY-Konsortium definierten XML-Format DTBook ausgezeichnet [1] – letzteres empfiehlt der Epub-Standard für stark strukturierte Inhalte wie Lehrbücher. Um die Darstellung und die Anpassung an Display- und Schriftgröße kümmert sich das Lesegerät. Gegenüber reinem HTML bietet der Epub-Standard den Vorteil, dass dort auch die Metadaten wie Autor und Titel, ein Inhaltsverzeichnis oder die Anzeigereihenfolge der verschiedenen Teile eines Buchs in XML-Dateien gespeichert sind.

Diese XML-Dateien lassen sich aus strukturierten Inhalten, etwa aus einem Content

Management System, automatisiert per Software erzeugen, aber auch relativ einfach von Hand erstellen. Es gibt eine Reihe von Programmen, die diverse Formate nach Epub wandeln – Calibre beispielsweise ist so ein Tool, das verschiedenste Formate von und nach Epub wandelt, E-Books anzeigt und die E-Book-Bibliothek verwaltet. Bei unseren Versuchen erzeugte die für Windows, Mac OS X und Linux verfügbare Software allerdings gelegentlich E-Books, die nicht alle Reader anzeigen konnten, und griff auch beim automatischen Erstellen des Inhaltsverzeichnisses ab und zu daneben.

Drei verschiedene Standards hat das IDPF definiert, die das Epub-Format festlegen: Die Open Publication Structure 2.0 (OPS [2]) beschreibt die Formatierung des Inhalts, das Open Packaging Format 2.0 (OPF [3]) definiert die Struktur und den Inhalt der XML-Dateien mit den Metadaten und das Open eBook Publication Structure Container Format 1.0 (OCF [4]) legt fest, wie die verschiedenen Dateien in einer Zip-komprimierten Container-Datei zusammengefasst werden. In der Regel ist es allerdings nicht nötig, diese Standards im Detail zu studieren: Die meisten E-Books kommen mit einem Standardgerüst aus.

Das Epub-Format unterstützt digitale Signaturen, das Verschlüsseln der Inhalte sowie Digital Rights Management (DRM). Damit eignet es sich auch für den Vertrieb kopiergeschützter E-Books – Online-Shops wie Libreka, die E-Book-Plattform des Börsenvereins des deutschen Buchhandels, Libri.de und Ciando bieten DRM-geschützte E-Books im Epub-Format an.

Prolog

Das Bauen eines eigenen E-Books im Epub-Format erfolgt in drei Schritten. Zunächst muss man die Inhalte in XHMTL formatieren, dann die erforderlichen XML-Dateien mit den Metadaten erzeugen und schließlich alles in ein standardkonformes E-Book zusammenpacken. All das lässt sich automatisiert per Software, aber auch in Handarbeit erledigen. Dabei helfen verschiedene Tools, die Sie – genau wie alle Dateien für das Beispiel-E-Book – über den Link am Ende des Artikels finden.

Wichtigstes Werkzeug für die ersten beiden Schritte ist ein XML-Editor, mindestens aber ein Editor, der Syntax-Highlighting für XML beherrscht. Wenn er zudem den XML-Code validieren kann, umso besser – der OPF-Standard enthält im Anhang ein Relax-NG-Schema für das wichtigste Metadaten-Format. Ansonsten hilft ein XML-Validator weiter, beispielsweise XMLStarlet für Unix und Linux. HTML Tidy bereinigt den „schmutzigen" HTML-Code, den man auf vielen Webseiten findet, und macht sauberes XHTML daraus.

Cextra ist ein nützliches Werkzeug, das den ersten Schritt vereinfacht, indem es die urheberrechtsfreien Bücher des Gutenberg-Projekts oder von Zeno.org herunterlädt und als Epub-taugliche XHTML-Datei speichert. Allerdings befindet sich das Java-Programm noch in einer frühen Entwicklungsphase; so haben wir bei unseren Versuchen sowohl Abstürze als auch Probleme mit der korrekten Umsetzung von Umlauten erlebt. Dennoch lohnt bei den Zeno- und Gutenberg-Büchern ein Versuch: Wenn es klappt, spart man sich unter Umständen eine Menge Mühe beim Nachbearbeiten des HTML-Codes aus dem Web.

Beim Überprüfen des fertigen E-Books leistet FBReader, ein schlichter Epub-Viewer für den Windows- und Linux-Desktop, gute Dienste, auch wenn das Programm bei der Interpretation von Stylesheets gelegentlich eigene Wege geht. Eine Alternative ist der Ebook-Viewer, der zu dem schon erwähnten Calibre gehört. Unverzichtbar ist der Epub-Checker von Adobe, der E-Books darauf prüft, ob sie die Standards einhalten. Das plattformunabhängige Java-Tool ist zwar mäkeliger, als es sein müsste – diverse von EpubCheck angemeckerte Bücher wurden auf verschiedenen Readern anstandslos angezeigt –, aber es ist der schnellste Weg, Fehler in der Epub-Struktur zu finden.

Ein interessantes Projekt ist eCub, das die Schritte zwei und drei – das Erstellen der XML-Dateien mit den Metadaten und das Verpacken in ein E-Book – in eine komfortable grafische Oberfläche packt. Zu vernünftigen Ergebnissen kommt man mit dem für verschiedene Betriebssysteme verfügbaren Tool allerdings nur, wenn die Inhalte bereits als XHTML aufbereitet sind. Kurz erwähnt sei hier noch Sigil, ein plattformunabhängiger Wysiwyg-Editor für E-Books. Die Qt-Software wirkt in der aktuellen Version 0.1.4 noch etwas unfertig; auch erlebten wir immer wieder Abstürze beim Öffnen oder Speichern von E-Books.

Gliederung

Ein E-Book im Epub-Format muss eine Reihe von Dateien enthalten. Der Grundaufbau sieht typischerweise so aus:

```
mimetype
META-INF/container.xml
OEBPS/inhalt.opf
OEBPS/toc.ncx
OEBPS/inhalt1.xhtml
OEBPS/style.css
OEBPS/...
```

Alle diese Dateien werden am Ende in einer Zip-Datei zusammengefasst – das ist dann schon das E-Book.

In der Datei mimetype steht lediglich eine Zeile mit dem Inhalt

```
application/epub+zip
```

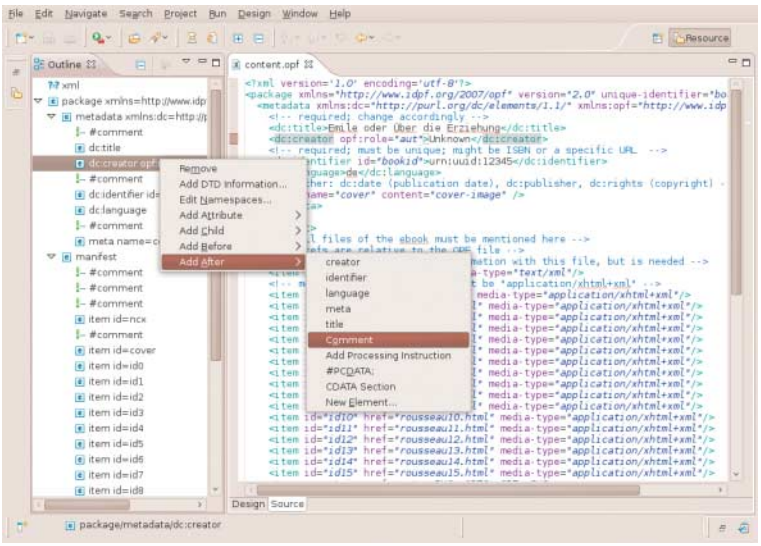
Die Datei darf keinen Zeilenumbruch am Ende enthalten und muss unkomprimiert am Anfang der Zip-Datei stehen – wie man das hinkriegt, dazu später mehr.

Die Datei container.xml im Unterverzeichnis META-INF ist der Einstiegspunkt in das E-Book. Sie enthält lediglich einen Eintrag, der an das konkrete E-Book anzupassen ist: das Element <rootfile> mit dem Pfad zu der zentralen Metadaten-Datei, dem OPF Package Document, relativ zum „Stammverzeichnis" des E-Books. Das kann dann so aussehen:

```
<?xml version="1.0"?>
<container version="1.0"
 xmlns="urn:oasis:names:tc:opendocument:xmlns:container">
  <rootfiles>
    <rootfile full-path="OEBPS/inhalt.opf"
     media-type="application/oebps-package+xml" />
  </rootfiles>
</container>
```

Falls das E-Book verschlüsselt, signiert oder DRM-geschützt ist, finden sich in META-INF weitere Dateien.

Für alle anderen Dateien im E-Book gilt, dass man ihre Namen und die Verzeichnisse, in denen sie gespeichert sind, frei wählen kann. Der Text kann auf beliebig viele XHTML- oder DTBook-Dateien aufgeteilt werden; es spricht beispielsweise nichts dagegen, jedes Kapitel eines Buches in eine eigene Datei zu packen. Als Konvention hat sich eingebürgert, das OPF Package Docu-



Wichtigstes Werkzeug zum Bauen eines E-Books ist ein XML-Editor, beispielsweise Eclipse.

ment und die dort referenzierten Dateien mit dem Inhalt in einem Verzeichnis mit dem Namen OEBPS (als Abkürzung für Open EBook Publication Structure) abzulegen.

Katalog

Das OPF Package Document, auf das die Datei META-INF/container.xml verweist, ist der zentrale Dreh- und Angelpunkt des E-Books. Die Datei muss neben einigen Metadaten die Namen aller Dateien enthalten, die zu dem E-Book gehören. Sie besteht aus vier Sektionen.

Die Sektion <metadata> enthält die im Dublin Core Metadata Element Set aufgeführten Metadaten [5]. Dazu gehören der Titel und Autor des Buchs (dc:title und dc:creator), seine Sprache (dc:language), das Datum (dc:date), der Verlag (dc:publisher), Schlagworte, die das Thema des E-Books beschreiben (dc:subject), und eine eindeutige Kennung (dc:identifier), bei Büchern beispielsweise die ISBN-Nummer der gedruckten Vorlage. Außerdem lassen sich hier beliebige andere Metadaten angeben. Die Elemente dc:title, dc:language und dc:identifier sind zwingend erforderlich.

Die zweite Sektion, <manifest>, führt alle Dateien auf, die zu dem E-Book gehören. Ihre Pfade sind relativ zum OPF Package Document, nicht etwa zum „Stammverzeichnis" des E-Books anzugeben. Zu der Liste der Dateien, die hier auftauchen, gehören die XHTML-Datei(en) mit dem Textinhalt, die daraus referenzierten Bilder, CSS-Stylesheets sowie eine Datei mit dem Inhaltsverzeichnis (Navigation Control file for XML, kurz NCX). Bei allen Dateien muss man den Mimetype angeben, und jede Datei erhält hier eine eindeutige ID.

Die dritte Sektion namens <spine> (zu Deutsch: Buchrücken) gibt die Reihenfolge vor, in der Reader die XHTML-Dateien anzeigen sollen, wenn man das E-Book von vorne nach hinten liest. Die Dateien werden dabei über ihre in der <manifest>-Sektion vergebene ID referenziert.

Während die ersten drei Sektionen vorhanden sein müssen, ist die vierte Sektion <guide> optional. Hier kann man einzelnen Teilen des E-Books eine strukturelle Bedeutung wie Cover (cover), Titelseite (title-page), Inhaltsverzeichnis (toc), Vorwort (foreword), Index (index), Liste der Illustrationen (loi) und Tabellen (toi) sowie erste „echte" Textseite (text) zuweisen. Die entsprechenden Stellen im Buch bezeichnet man durch href-Links, die auch einen Anker innerhalb einer Datei ansprungen können. Nicht alle E-Book-Reader werten diese Sektion aus.

Stoffsammlung

Bevor es an das Erstellen der Metadaten geht, benötigt man allerdings erst einmal Daten, sprich: XHTML-formatierten Inhalt. Den findet man beispielsweise beim Projekt Gutenberg; eine noch etwas größere Sammlung mit Literatur, deren Urheberrecht abgelaufen ist, bietet Zeno.org. Dass das Urheberrecht abgelaufen ist, heißt allerdings nicht, dass man mit Büchern aus diesen Quellen ganz nach Belieben verfahren darf: Zeno.org nimmt ein Datenbank-Copyright auf die Zusammenstellung der Werke für sich in Anspruch und erlaubt es derzeit lediglich, einzelne Bücher herunterzuladen und zum privaten Gebrauch in ein E-Book umzuwandeln. Das Projekt Gutenberg erlaubt auch die kostenlose Weitergabe von E-Book-Versionen seiner Inhalte.

Als Ausgangspunkt soll hier Tucholskys „Schloss Gripsholm" vom Gutenberg-Projekt dienen [6], ein kleines Büchlein mit vier Kapiteln. Man kann die vier Kapitel einfach nacheinander aus dem Browser speichern, dabei landen zur Darstellung benötigte zusätzliche Dateien wie CSS-Stylesheets in Unterverzeichnissen.

Als Inhalt ist in E-Books fast alles erlaubt, was sich in XHTML 1.1 ausdrücken lässt – mit einigen Ausnahmen: <img>-Tags dürfen im „src"-Attribut nur lokale Bilder referenzieren und müssen ein „alt"-Attribut haben. JavaScript ist zwar theoretisch erlaubt; allerdings muss ein Reader das E-Book auch korrekt darstellen können, wenn er den JavaScript-Code komplett ignoriert, wie es der Standard empfiehlt (und es auch nahezu alle Reader tun). Links auf im E-Book enthaltene Dateien, die nicht im Manifest der OPF-Datei aufgeführt werden, sind erlaubt, Lesegeräte müssen sie jedoch nicht berücksichtigen. Externe Links sind möglich, können aber natürlich nur funktionieren, wenn der E-Book-Reader aufs Web zugreifen kann – was bei den derzeit in Deutschland erhältlichen Epub-fähigen Lesegeräten nicht der Fall ist. Zudem führt der OPS-Standard einige XHTML-Erweiterungen ein, die aber nur in Spezialfällen interessant sind.

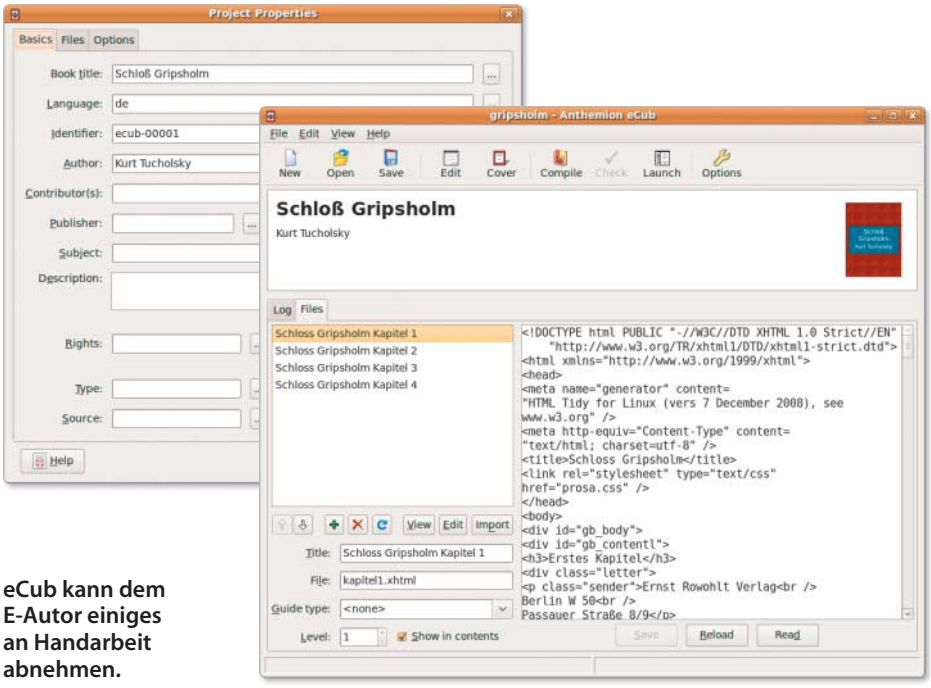
Die XHTML-Dateien müssen in Unicode (UTF-8 oder UTF-16) kodiert sein, andere Zeichensätze wie ISO8859-1 (latin-1) sind nicht erlaubt. Einige Lesegeräte haben Probleme damit, XHTML-Dateien über einer gewissen Größe anzuzeigen; der Sony-Reader PRS-505 beispielsweise streikt bei E-Books, in denen einzelne XHTML-Dateien die Größe von 300 KByte erreichen. Sollten einzelne Dateien größer sein, muss man sie aufteilen.

Reinschrift

Da die Online-Anzeige des Buches bei Gutenberg von Navigationselementen und Werbung umgeben ist, die in dem E-Book nichts zu suchen haben, muss in den vier HTML-Dateien zunächst aufgeräumt werden: Diverse Gutenberg-spezifische Informationen, die Navigation, Anzeigen, externe Links und sämtlicher JavaScript-Code müssen weg.

Aus dem Header benötigt man lediglich die Textkodierung, den <title>-Eintrag (am besten geändert auf den Titel des Buches) und den Verweis auf das Stylesheet prosa.css. Diese Datei, nach dem Download im Unterverzeichnis zur HTML-Datei zu finden, kopiert man am besten gleich in dasselbe Verzeichnis wie die zugehörige HMTL-Datei und passt den Link auf das Stylesheet entsprechend an. Wer die Darstellung des Textes anpassen will, kann sich in dem Stylesheet austoben.

Eine Änderung im Stylesheet ist allerdings auf jeden Fall nötig: Gutenberg definiert die Schriftgrößen in der Einheit Pixel mit einer Größe von 12px für den Text – viel zu klein für Lesegeräte mit einer Auflösung



eCub kann dem E-Autor einiges an Handarbeit abnehmen.

von 150 bis 200 dpi, zumal sich die meisten Lesegeräte sklavisch daran halten und die Einstellung einer anderen Schriftgröße verweigern. Die Lösung ist einfach: alle Fontsize-Angaben mit px-Angaben im Stylesheet entfernen.

Im Body kann bis zur ersten Zeile des Textes (die Überschrift des Kapitels in <h3>-Tags) bis auf die beiden Tags gb_body und gb_contentl alles gelöscht werden. Lediglich einige der vor dem Text im Body untergebrachten <gb_meta>-Einträge mit den Gutenberg-Metadaten lassen sich später für die Metadaten des E-Books wiederverwenden; man sollte diesen Code-Block daher irgendwo aufheben. Auch sämtlicher HTML-Code hinter dem Ende des Textes kann weg.

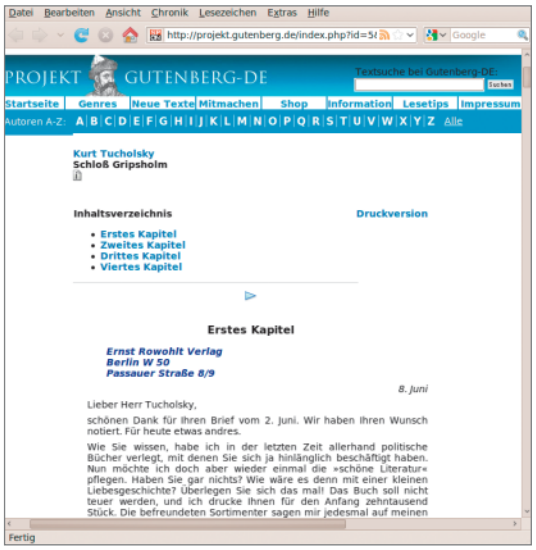
Im <html>-Tag am Anfang der Dateien fehlt noch ein Verweis auf den XHTML-Namensraum:

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

Die HTML-Dateien sollten jetzt ungefähr so aussehen:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/loose.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type"
 content="text/html; charset=ISO-8859-1">
<title>Schloss Gripsholm</title>
<link rel="stylesheet" type="text/css"
      href="prosa.css">
</head><body>
<div id="gb_body"><div id="gb_contentl">
<h3>Erstes Kapitel</h3>
...
... Gute Nacht, Prinzessin.</p></div></div>
</body></html>
```

„Schloss Gripsholm“ liefert den Inhalt für das E-Book.

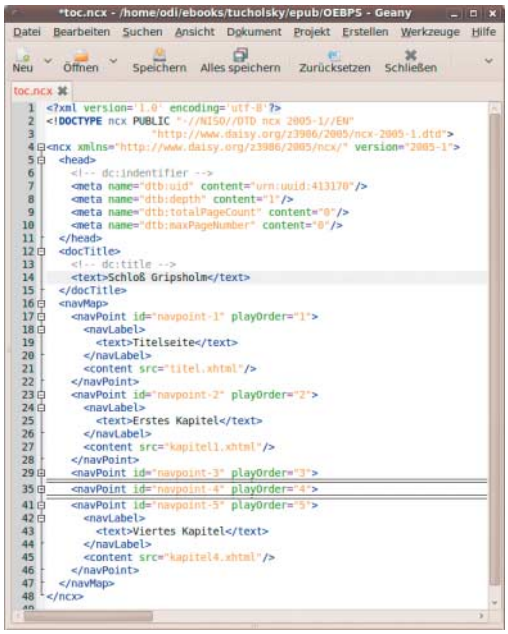


Nun wandelt HMTL Tidy die aufgeräumten HTML-Dateien von Gutenberg in sauberen XHTML-Code um und setzt dabei gleich die Zeichenkodierung auf das vom E-Book-Standard geforderte UTF-8 um:

```
tidy -clean --input-encoding latin1 --output-encoding ↗
     utf8 -asxhtml -o kapitel1.xhtml kapitel1.html
```

Wenn das Tool lediglich ein paar Warnungen, aber keine Fehler meldet und am Ende von „XHTML 1.0 Strict“ spricht, hat alles funktioniert. Im dritten und vierten Kapitel ist in der HTML-Datei noch eine Änderung von Hand nötig: Der Text innerhalb der <blockquote>-Tags muss in und eingeschlossen werden, damit das XHTML valide ist. Die von HTML Tidy erzeugten XHTML-Dateien sollten in einem Browser ungefähr so aussehen wie bei Gutenberg.

Will man das E-Book richtig schön machen, erstellt man noch eine Titelseite, natürlich ebenfalls in XHTML. Eine der Dateien mit den Kapiteln kann als Vorlage dienen. Ein



Die Datei toc.ncx liefert die Navigation im E-Book.

Motiv für die Titelseite, etwa ein frei verwendbares Bild des titelgebenden Schlosses, findet man beispielsweise auf Wikimedia Commons. Es lässt sich über das übliche <img>-Tag einbinden. Erlaubte Formate sind JPEG, PNG, GIF und das Vektorformat SVG.

Erzählstruktur

Jetzt geht es an das Erstellen des OPF Package Document. An den Anfang gehören die Zeilen

```
<?xml version='1.0' encoding='utf-8'?>
<package xmlns="http://www.idpf.org/2007/opf"
 version="2.0" unique-identifier="bookid">
```

Die Metadaten-Sektion ist schnell mit den Angaben aus den Gutenberg-Metadaten befüllt; als <identifier> kann die Artikelnummer von Gutenberg dienen, die am Anfang des HMTL-Codes als spArticleID steht.

```
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:opf="http://www.idpf.org/2007/opf">
  <dc:title>Schloß Gripsholm</dc:title>
  <dc:creator>Kurt Tucholsky</dc:creator>
  <dc:language>de</dc:language>
  <dc:identifier id="bookid">urn:uuid:413170</dc:identifier>
  <dc:publisher>Gutenberg-Projekt</dc:publisher>
  <dc:date>1931</dc:date>
</metadata>
```

Das Manifest ordnet den zum E-Book gehörenden Dateien eine ID zu und legt ihren Dateityp fest. Die am Anfang der Sektion genannte Datei toc.ncx enthält ein Inhaltsverzeichnis; sie ist zwingend erforderlich und muss die ID „ncx“ haben.

```
<manifest>
  <item id="ncx" href="toc.ncx"
   media-type="text/xml"/>
  <item id="titel" href="titel.xhtml"
   media-type="application/xhtml+xml"/>
  <item id="kap1" href="kapitel1.xhtml"
   media-type="application/xhtml+xml"/>
  <item id="kap2" href="kapitel2.xhtml"
   media-type="application/xhtml+xml"/>
  <item id="kap3" href="kapitel3.xhtml"
   media-type="application/xhtml+xml"/>
  <item id="kap4" href="kapitel4.xhtml"
   media-type="application/xhtml+xml"/>
  <item id="titelbild" href="gripsholm.jpg"
   media-type="image/jpeg"/>
  <item id="css" href="prosa.css"
   media-type="text/css"/>
</manifest>
```

Der OPF-Standard kennt einen Mechanismus, über den man Inhalte auch in solchen Formaten in ein E-Book packen kann, die der Epub-Standard nicht vorsieht. Für solche Bestandteile ist es zwingend erforderlich, den Inhalt auch in einem Epub-kompatiblen Format als Fallback aufzunehmen. Dem Reader steht es dann frei, eines der anderen Formate anzuzeigen. Das kann dann in der Manifest-Sektion etwa so aussehen:

```
<item id="kap1" href="kapitel1.pdf"
 media-type="application/pdf" fallback="fb1" />
<item id="fb1" ref="kapitel1.xhtml"
 media-type="application/xhtml+xml" />
```

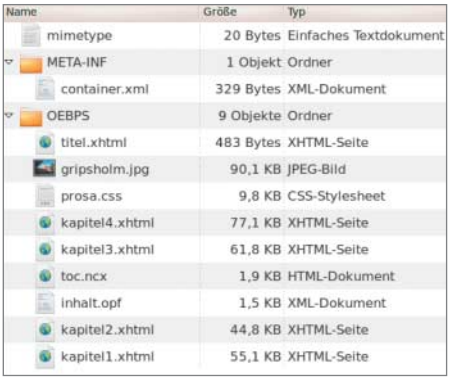
Reader dürfen in diesem Fall die PDF-Version anzeigen, können aber auch auf das XHTML-Format zurückfallen. Da der Tucholsky nur in XHTML vorliegt, kommt das E-Book ohne Fallbacks aus.

Fehlt nur noch die Lesereihenfolge mit dem schließenden <package>-Tag:

```
<spine toc="ncx">
  <itemref idref="titel">
  <itemref idref="kap1"/>
  <itemref idref="kap2"/>
  <itemref idref="kap3"/>
  <itemref idref="kap4"/>
</spine>
</package>
```

Mit dem Attribut linear="no"/ im <itemref>-Tag kann man einzelne Dateien als zusätzliche Teile kennzeichnen, die dann beim seitenweisen Durchblättern des Buches nicht angezeigt werden. Damit sie überhaupt zu sehen sind, muss man solche Dateien an anderen Stellen im Inhalt des E-Books verlinken.

Die komplette OPF-Datei finden Sie über den Link am Ende des Artikels.



Ein leeres Verzeichnis nimmt alle Dateien des E-Books auf.

Inhaltsverzeichnis

Für das bereits erwähnte Inhaltsverzeichnis, das eine Navigation im Buch über das lineare Lesen hinaus erlaubt, greift der Epub-Standard auf einen Teil des ANSI/NISO-Standards Z39.86-2005 (Specifications for the Digital Talking Book) des DAISY-Konsortiums zurück [1]. Dieser Standard beschreibt in seinem Kapitel acht den Aufbau des Navigation Control File (NCX), ebenfalls eine XML-Datei. Der Anfang der Datei verweist auf den DAISY-Standard:

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE ncx PUBLIC "-//NISO//DTD ncx 2005-1//EN"
"http://www.daisy.org/z3986/2005/ncx-2005-1.dtd">
<ncx xmlns="http://www.daisy.org/z3986/2005/ncx/"
version="2005-1">
```

Der Header muss vier <meta>-Tags mit den folgenden Attributen enthalten: dtb:uid erhält denselben Wert wie <dc:identifier> in der OPF-Datei. dtb:depth gibt die Verschachtelungstiefe des Inhaltsverzeichnisses an, dtb:totalPageCount und dtb:maxPageNumber sind bei einem E-Book irrelevant und können auf null gesetzt werden. Außerdem ist ein <doctitle>-Tag erforderlich, das den Titel des Buchs enthält. Bei dem Tucholsky sieht das dann so aus:

```
<head>
  <meta name="dtb:uid" content="urn:uuid:413170"/>
  <meta name="dtb:depth" content="1"/>
  <meta name="dtb:totalPageCount" content="0"/>
  <meta name="dtb:maxPageNumber" content="0"/>
</head>
<docTitle>
  <text>Schloß Gripsholm</text>
</docTitle>
```

Das Tag <navMap> leitet das eigentliche Inhaltsverzeichnis ein. NCX erlaubt das Erstellen komplexer, hierarchischer Inhaltsverzeichnisse, für den Tucholsky reichen jedoch fünf Einträge für die Titelseite und die vier Kapitel. Für jeden ist ein <navPoint>-Tag nach folgendem Aufbau erforderlich:

```
<navPoint id="navpoint-1" playOrder="1">
  <navLabel>
  <text>Ttitelseite</text>
  </navLabel>
  <content src="titel.xhtml"/>
</navPoint>
```

Die ID kann man frei wählen, playOrder gibt wie die Spine-Sektion der OPF-Datei die Lesereihenfolge an. Der Unterschied: <spine> listet lediglich die vorhandenen Dateien auf, während die NCX-Datei auch zusätzliche Einträge enthalten kann, die im <content>-Tag auf einen Anker innerhalb einer Datei verweisen, sodass sich in der NCX-Datei ein feiner ge-

gliedertes Inhaltsverzeichnis bauen lässt. Das <navLabel>-Tag enthält lediglich einen <text>-Eintrag mit der Zeile, die im Inhaltsverzeichnis angezeigt werden soll. Am Ende der Datei muss dann noch </ncx> stehen.

Für ein Inhaltsverzeichnis mit zwei Gliederungsebenen setzt man das Attribut „dtb:depth“ auf 2; die geschachtelten <navPoint>-Tags sehen dann beispielsweise so aus:

```
<navMap>
  <navPoint id="nav-1" playOrder="1">
    <navLabel>
    <text>Kapitel 1</text>
    </navLabel>
    <content src="kap1.xhtml" />
    <navPoint id="nav-11" playOrder="2">
      <navLabel>
      <text>Kapitel 1, Absatz 1</text>
      </navLabel>
      <content src="kap1.xhtml#nav-11" />
    </navPoint>
    <navPoint id="nav-12" playOrder="3">
      <navLabel>
      <text>Kapitel 1, Absatz 2</text>
      </navLabel>
      <content src="kap1.xhtml#nav-12" />
    </navPoint>
  </navPoint>
  <navPoint id="nav-2" playOrder="4">
    <navLabel>
    <text>Kapitel 2</text>
    </navLabel>
    ...
  </navPoint>
</navMap>
```

Buchbinder

Nachdem jetzt alle Bestandteile zusammen sind, kann man die Dateien in ein leeres Verzeichnis packen, das die Datei mimetype und die beiden Unterverzeichnisse META-INF und OEBPS enthält. Die bereits erwähnte Datei container.xml mit dem Verweis auf das OPF-File gehört nach META-INF, die XHTML-Dateien, das Stylesheet, das Titelbild, die OPF- und die NCX-Datei wandern nach OEBPS.

Der Befehl

```
zip -0X gripsholm.epub mimetype
```

erzeugt zunächst ein Zip-Archiv, das lediglich die Datei mimetype unkomprimiert (Option -0) und ohne zusätzliche Dateiattribute (-X) enthält.

```
zip -Xr9D $1 *
```

packt anschließend alle anderen Dateien dazu – komprimiert (-9), rekursiv über alle Unterverzeichnisse (-r) und ohne eigene Einträge für die Unterverzeichnisse OEBPS und META-INF (-D).

Wer auf Nummer sicher gehen will, lässt jetzt Adobes Epub-Checker mit

```
java -jar epubcheck-1.0.3.jar gripsholm.epub
```

über die Datei laufen. Achten sollte man vor allem auf Fehlermeldungen, die die Struktur des E-Books betreffen. Wenn das Tool lediglich wegen nicht erlaubter Attribute in eigentlich validen XHTML-Dateien meckert, ist das E-Book trotzdem in Ordnung.

Nun kann man das E-Book auf ein Lesegerät kopieren – und mit der Lektüre beginnen.

(odi)

Literatur

[1] DAISY-Standard ANSI/NISO Z39.86: www.daisy.org/z3986/

[2] Open Publication Structure 2.0 (OPS): www.openebook.org/2007/ops/OPS_2.0_final_spec.html

[3] Open Packaging Format 2.0 (OPF): www.openebook.org/2007/opf/OPF_2.0_final_spec.html

[4] OEBPS Container Format 1.0 (OCF): www.openebook.org/ocf/ocf1.0/download/ocf10.htm

[5] Dublin Core Metadata Element Set: www.dublincore.org/documents/dces

[6] Schloss Gripsholm: http://projekt.gutenberg.de/index.php?id=5&xid=2896&kapitel=1

www.ct.de/0925146 c't