



Dr. Victor  
Wang

# E-Books mit ePUB

## Von Word zum E-Book mit XML

# Erstellung eines E-Books mit dem Mobipocket Creator

Dieses Kapitel soll einen ersten Einstieg in die praktische Erstellung eines E-Books geben. Als Zielformat wird das im vorherigen Kapitel erwähnte Mobipocket-Format gewählt, das aus drei Gründen für den Anfang besonders geeignet ist:

1. Zum einen ist Mobipocket wie ePUB ein Datenformat, das ebenfalls auf XML-Standards aufsetzt und sich deshalb anbietet, da es bereits einige wesentliche Konzepte enthält, die später noch für ePUB benötigt werden. Bereits erwähnt wurde außerdem, dass das Mobipocket-Format auf dem Open-eBook-Format, also der unmittelbaren Vorgängerversion von ePUB, aufsetzt.
2. Mobipocket kann neben ePUB als eines der verbreiteteren E-Book-Formate am Markt gelten (<http://www.mobipocket.com>). Es kann außerdem von allen Kindle-Geräten gelesen werden.
3. Mit dem Mobipocket Creator liegt ein Werkzeug vor, das mit einer verständlichen Benutzeroberfläche E-Books ohne weitere spezielle Kenntnisse der dahinterliegenden XML-Datenstrukturen erzeugt.

Der im Folgenden dargestellte Ablauf einer Mobipocket-Produktion sieht schematisch so aus: Zunächst müssen die Quelldaten (Texte, Bilder) mit dem Mobipocket Creator in einem Projekt zusammengeführt werden; das Projekt wird dann geprüft und in das MOBI-Endformat kompiliert; zum Testen der so erzeugten MOBI-Datei kommt schließlich der Mobipocket Reader zum Einsatz.

## 3.1 Installation

Die beiden erwähnten Softwarekomponenten, der Mobipocket Creator und Reader, sind auf der Entwickler-Website von Mobipocket.com verfügbar:<sup>1</sup>

- **Der Mobipocket Creator:** Die Entwicklungsumgebung gibt es in zwei Varianten, der Home Edition sowie der Publisher Edition; während die erstgenannte für den privaten Bedarf gedacht ist, unterstützt die Publisher-Fassung die Vergabe von DRM und Aufnahme von bibliografischen Angaben. Der Creator bietet eine grafische Benutzeroberfläche und zielt insgesamt auf den technisch weniger versierten Anwender, der sich mit wenigen Klicks ein E-Book zusammenstellen möchte. Leider ist der Creator nur für Windows-Systeme verfügbar.
- **Der Mobipocket Reader:** Um sich einen Eindruck des fertigen E-Books zu machen, wird eine Anwendung benötigt, die das Ergebnis möglichst nahe an den verfügbaren Hardware-Readern anzeigt. Neben der Anzeigefunktion übernimmt der Reader außerdem die Aufgabe der Titelverwaltung: Er merkt sich alle geöffneten MOBI-Titel, vergleichbar einer digitalen Bibliothek. Der Reader ist neben der hier verwendeten Windows-Fassung auch für diverse mobile Endgeräte verfügbar.<sup>2</sup>

Die Installation der Software ist unspektakulär: Nach dem Start der `creator.msi` muss nach der Bestätigung der Lizenzbestimmungen lediglich ausgewählt werden, welche Variante installiert werden soll. Für die Testzwecke dieses Kapitels wurde die Creator Publisher Edition verwendet. Die Installation des Readers (`mobi reader setup.msi`) weist keine wichtigen Optionen auf. Nach der Wahl des Installationsordners werden die Programmdateien von bescheidenem Umfang (der Creator umfasst 12 MB, der Reader 11 MB) kopiert. Danach kann der Creator wie der Reader direkt ohne Neustart von Windows gestartet werden.

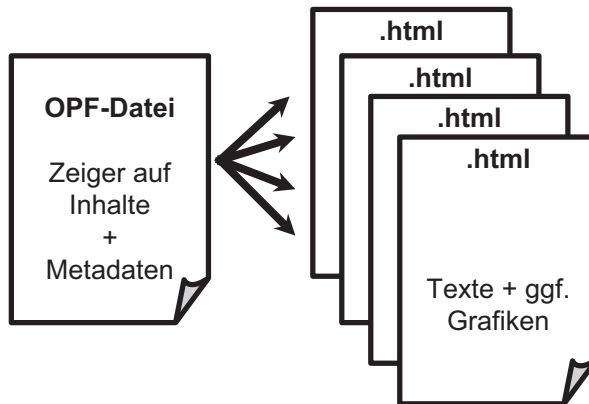
## 3.2 Mobipocket und das mobi-Format<sup>3</sup>

Auf der Entwicklerseite von Mobipocket wird das Mobipocket-Format als »ein Binärformat zur Verteilung von E-Books« bezeichnet. Ursprünglich ist es wohl aus dem PalmDOC-Format entstanden, dem HTML-ähnliche Elemente hinzugefügt wurden. Daher kommt auch die leicht irritierende Dateiendung `.prc` (Palm Resource Compiler file), die die meisten Mobipocket-E-Books tragen, da das Palm OS nur diese akzeptiert.

Trotzdem setzt das Mobipocket-Format auf aktuelle XML- bzw. HTML-Standards in den Quelldaten auf, genauer auf das bereits in Abschnitt 2.1.1 erwähnte Open-E-Book-Format, das nur geringfügig erweitert wurde.

Schematisch kann man sich die Quelldaten wie folgt (Abbildung 3.1) vorstellen:

- Die eigentlichen Inhalte liegen als HTML-Dateien (genauer: XHTML-Dateien) vor.
- Die so genannte OPF-Datei (Open E-Book Publication Format) referenziert alle im E-Book aufzunehmenden Inhaltsdateien.
- Die für die Publikation ebenfalls nötigen Metadaten, wie beispielsweise Titel, Autor, Verlag, sind ebenfalls in dieser OPF-Datei enthalten.



**Abbildung 3.1:** Zusammenspiel der Mobipocket-Quelldaten

Die OPF-Datei ist demnach die zentrale Steuerungsdatei für die E-Book-Produktion, sie liegt folglich auch dem Mobipocket Creator zugrunde bzw. wird von diesem als Quelldatenformat erzeugt. Anders ausgedrückt, übernimmt der Creator damit quasi die Funktion eines Editors für die im XML-Format vorliegende OPF-Datei.

Der wesentliche Schritt in der Produktion besteht jedoch nun darin, dass der Creator aus den Quelldaten das gewünschte Zielformat Mobipocket erzeugt. Im Zuge dieser Kompilierung werden die folgenden Schritte durchlaufen:

- Alle Inhalte werden in eine einzige Datei zusammengepackt – das E-Book soll ja später als eine Datei versandt werden
- Optimierung der gepackten Datei zwecks verbesserter Verarbeitung auf den Endgeräten
- Komprimierung der Inhalte, damit der Umfang des späteren E-Books handlich bleibt
- Optionale Indizierung (soweit eine entsprechende Suche im späteren E-Book gewünscht wird)

- Optionale Verschlüsselung, wenn das E-Book später mit einem digitalen Kopierschutz (DRM) versehen werden soll

Als Ergebnis schreibt der Creator dann eine Binärdatei mit der Dateierendung .mobi oder .prc, die dann auf den Endgeräten oder dem Reader dargestellt werden kann.

Die Qualität des späteren Mobipocket-E-Books hängt stark von den zugrunde liegenden Inhaltsdateien, den genannten HTML-Daten, ab. Doch liegen diese häufig als Ausgangsdaten noch gar nicht vor, sondern stattdessen die üblichen Text- oder Seitenformate wie Word, RTF oder PDF. Aus diesen muss erst einmal HTML erzeugt und eventuell noch nachbearbeitet werden. Spätestens wenn dieser zusätzliche Schritt notwendig wird, muss die bequeme Oberfläche des Creators verlassen und in die häufig mühsame Datenbearbeitung eingestiegen werden. Da Mobipocket an dieser Stelle nur als Einstiegsbeispiel dient und Word als Datenbasis noch ausführlicher behandelt werden wird, sei hier nur kurz auf die möglichen Abläufe hingewiesen<sup>4</sup>:

Ausgangsdaten	OPF	HTML	Bewertung
Open E-Book (IDPF 1.0) oder ePUB (IDPF 2.0)	Besteht in diesen Formaten und kann direkt eingelesen werden	Besteht und kann unverändert übernommen werden	Bester Fall ohne Nacharbeit
HTML-Daten	Wird im Creator nach Hinzufügen der HTML-Daten erzeugt	Besteht und kann meist unverändert übernommen werden	Abhängig von der Herkunft der HTML-Daten muss ggf. nachgearbeitet werden
XML-Daten	Wird im Creator nach Hinzufügen der Ausgangsdaten erzeugt	HTML wird aus den XML-Daten durch ein XSLT-Skript erzeugt; das XSLT-Skript wird im Creator mitverwaltet und gestartet	Aufwand entsteht eventuell durch die Erstellung des XSLT-Skripts; die Qualität der HTML-Daten hängt von der Struktur der XML-Daten ab, ist aber in der Regel recht gut
Word/RTF	Wird im Creator nach Hinzufügen der Ausgangsdaten erzeugt	HTML wird zwar durch den Creator erzeugt, das Ergebnis ist jedoch in der Regel noch nicht befriedigend	Aufwand durch Nachbearbeitung der erzeugten HTML-Daten

Ausgangsdaten	OPF	HTML	Bewertung
PDF	Wird im Creator nach Hinzufügen der Ausgangsdaten erzeugt	HTML wird zwar durch den Creator erzeugt, das Ergebnis ist jedoch in der Regel noch nicht befriedigend	Aufwand durch erhebliche Nachbearbeitung der erzeugten HTML-Daten, da durch das seitenorientierte PDF-Format sehr viele Kontextinformationen verloren gegangen sind

## 3.3 Der Praxistest – »Max und Moritz« goes Mobipocket

### 3.3.1 »Max und Moritz« in XHTML

Nachdem nun die wesentlichen Werkzeuge zur Verfügung stehen und die Grundlagen der Quell- und Zieldaten bekannt sind, wollen wir auf dieser Basis ein echtes Mobipocket-E-Book produzieren.

Als Testobjekt wollen wir den Kinderbuch-Klassiker »Max und Moritz« von Wilhelm Busch auswählen. Die Streiche der beiden Lausbuben von 1865 sind hinreichend bekannt, außerdem kurz genug, um hier als Illustrationsbeispiel zu dienen. Die Knittelreime und Strichzeichnungen eignen sich außerdem bestens für die Anzeige in einem E-Book-Reader. Ein weiterer Vorteil ist, dass der Text digital verfügbar, urheberrechtsfrei und daher im Internet leicht zu finden ist. Die Darstellung stützt sich im Folgenden auf den Text von Wikisource ([http://de.wikisource.org/wiki/Max\\_und\\_Moritz](http://de.wikisource.org/wiki/Max_und_Moritz)), einer der freien, online zugänglichen Quelltextsammlungen<sup>5</sup>.

Betrachten wir das Vorwort und den ersten Streich, dem bekanntlich die Hühner der Witwe Bolte zum Opfer fallen. Die folgende XHTML-Datei wurde auf der Basis der Textdaten von Wikisource erzeugt und zeigt den idealtypischen Aufbau einer XHTML-Datei, die als Quelldatei der Mobipocket-Produktion dienen soll:

```

<?xml version="1.0" encoding="UTF-8"?> ❶
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"> ❷
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de"> ❸
  <head> ❹
    <meta http-equiv="Content-Type" content="text/html;
      charset=UTF-8" />

```

```

<title>Max und Moritz</title>
</head>
<body>
  <h1>Max und Moritz</h1>
  <div class="untertitel">Eine Bubengeschichte in sieben
    Streichen</div>
  <div class="jahr">(1865)</div>
  <div class="autor">von Wilhelm Busch</div>
  <h2>Vorwort</h2>
  <p>Ach, was muss man oft von bösen</p>
  <p>Kindern hören oder lesen!!</p>
  <p>Wie zum Beispiel hier von diesen,</p>
  <p>Welche Max und Moritz hießen;</p>
  <div></div>
  <p>Die, anstatt durch weise Lehren</p>
  <p>Sich zum Guten zu bekehren,</p>
  <p>Oftmals noch darüber lachten</p>
  <p>Und sich heimlich lustig machten. -</p>
  <p>- Ja, zur Übeltätigkeit,</p>
  <p>Ja, dazu ist man bereit! -</p>
  <p>. . .</p>
  <h2>Erster Streich</h2>
  <p>Mancher gibt sich viele Müh'</p>
  <p>Mit dem lieben Federvieh;</p>
  <p>. . .</p>
</body>
</html>

```

**Listing 3.1:** XHTML-Datei des ersten Streiches von Max und Moritz

Schauen wir uns die Struktur von XHTML an diesem Beispiel genauer an:

❶ Zunächst handelt es sich um eine echte XML-Datei, was die XML-Deklaration in der ersten Zeile ausweist.

❷ Es folgt dann die Festlegung des DOCTYPE, des so genannten Dokumenttyps; der DOCTYPE legt das Wurzelement fest und verweist auf das Regelwerk einer Document Type Definition (DTD), die dem vorliegenden XML-Dokument zugrunde liegt. Im Fall unseres Max-und-Moritz-Dokuments lautet das Wurzeldokument also <html>, die zugrunde liegende DTD heißt `xhtml1-strict.dtd` und ist im Internet an der angegebenen Adresse zu finden; zusätzlich wird der Public-Identifier festgelegt, mit dem diese DTD benannt wurde. Die DOCTYPE-Deklaration, die für normale HTML-Dateien entbehrlich ist, wird von der XHTML-Spezifikation zwingend gefordert.

❸ Hier steht nun das eigentliche Wurzelement `<html>`, das im darüber stehenden DOCTYPE definiert wurde; anders als in HTML muss in XHTML ein fester Namensraum mittels `xmlns="http://www.w3.org/1999/xhtml"` (»xmlns« legt den so genannten Default-Namensraum fest) angegeben werden; das Konzept von Namensräumen ist ein wichtiges XML-Konstrukt, das die Mischung von XML-Inhalten unterschiedlicher Herkunft, zum Beispiel unterschiedlicher Strukturen ermöglicht. Damit könnten beispielsweise XHTML-Fragmente in eine eigene XML-Struktur eingebettet werden; der angegebene Namensraum gilt für alle weiteren Kind-Knoten des XHTML-Dokuments und muss daher nicht bei jedem Element wiederholt werden. Das Attribut `xml:lang="de"` legt schließlich fest, dass das vorliegende Dokument in deutscher Sprache verfasst wurde.

❹ Der nun folgende HTML-Kopf `<head>` entspricht wieder ganz HTML: Das `<meta>`-Element wird als leeres Element XML-konform mit einem schließenden Slash beendet. Der Dokumenttitel steht in `<title>`.

❺ `<body>` enthält nun den eigentlichen Inhalt der XHTML-Datei; aus Umfangsgründen wurde im Beispiel der erste Streich stark gekürzt.

❻ Der Inhalt beginnt mit einer Überschrift der ersten Ebene `<h1>`, in der der Werktitel steht. Es folgen Untertitel, Autor und Erscheinungsjahr in einfachen Absätzen `<p>`.

❼ Die eigentlichen Überschriften, hier nur die des Vorwortes und des ersten Streiches, werden mit `<h2>`, also einer Überschrift der zweiten Ebene, markiert.

❽ Von den zahlreichen Abbildungen im Original wird hier nur die erste Grafik, die Max und Moritz zeigt, aufgenommen. Das Bild-Element `<img>` verweist im Attribut `src` auf die Quelle der Grafik, einer JPEG-Datei. Das Quellverzeichnis ist in diesem Fall relativ angelegt, die Grafikdateien liegen im selben Verzeichnis wie das vorliegende Quelldokument. Das zweite Attribut `alt`, das einen Alternativtext bei Anzeigeproblemen der Grafik enthält, wird von XHTML im Gegensatz zu HTML gefordert.

Diese Datei verwenden wir nun in der Folge für unsere Mobipocket-Produktion.

### 3.3.2 Anlage des Projektes im Mobipocket Creator und erster Schnelltest

Wir starten dazu zunächst den installierten Mobipocket Creator und öffnen dort eine neue, leere Publikation (»blank publication«), der wir in der nächsten Dialogbox den Titel »Max und Moritz goes Mobipocket« geben. Sobald dieser festgelegt ist, kommen wir zum zentralen Projektfenster (Abbildung 3.2).

### Kapitel 3

## Erstellung eines E-Books mit dem Mobipocket Creator



Abbildung 3.2: Verwaltungsoberfläche im Mobipocket Creator

Wir beginnen damit, die Publikationsdateien, die XHTML-Datei sowie die zugehörigen Grafikdateien in das Projekt per Drag&Drop einzufügen. In der Thumbnail-Ansicht sieht dann die Liste der Publikationsdateien wie folgt aus (Abbildung 3.3).

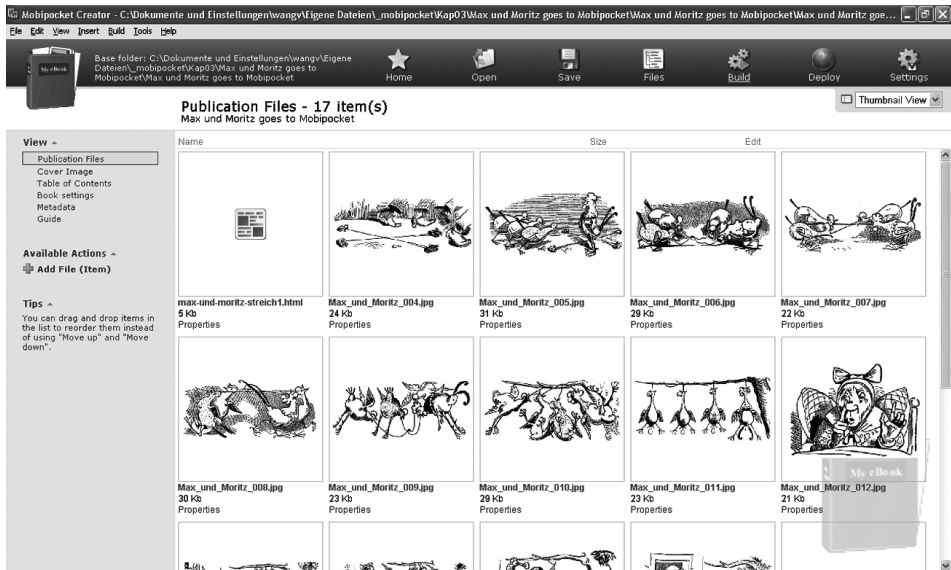


Abbildung 3.3: Thumbnail-Ansicht im Mobipocket Creator

Bevor wir uns nun die einzelnen Bestandteile des Projektes und der dahinter liegenden Mobipocket-Daten genauer ansehen, testen wir den Produktionsablauf, denn bereits mit den vorliegenden Daten lässt sich ein echtes E-Book erzeugen. Über das Menü »Build« kommen wir zur Dialogmaske, die die Kompilierung steuert. Wir lassen die Einstellungen für Kompression und Verschlüsselung unverändert und klicken auf den Button »Build«. Die Kompilierung läuft rasch durch und in der abschließenden »Build finished«-Maske wählen wir nun die Ansichtsoption »Preview with the Mobipocket Reader for PC«. Ein letzter Klick auf OK startet nun den Mobipocket Reader und öffnet unser frisch gebackenes Max-und-Moritz-E-Book (Abbildung 3.4).

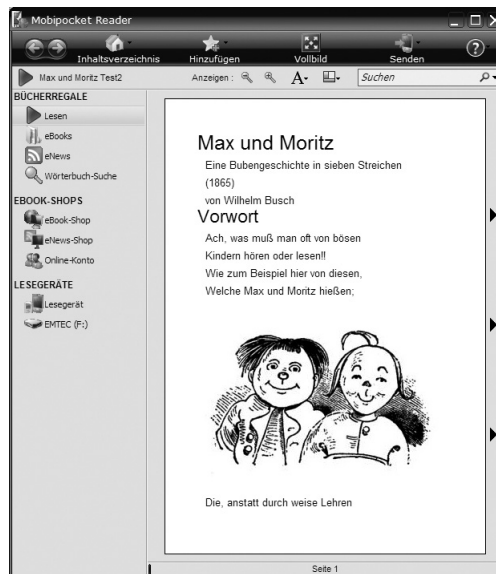


Abbildung 3.4: Ergebnis des Schnelltests im Mobipocket Reader

### 3.3.3 Verfeinerung der Mobipocket-Produktion

Um das E-Book publikationstauglich fertigzustellen, bietet der Creator die folgenden Möglichkeiten:

- Hinzufügen eines Cover-Bildes, das beim Start des E-Books im Reader angezeigt wird
- Hinzufügen eines Inhaltsverzeichnisses, das die Gliederung des Inhalts anzeigt und auf die entsprechenden Passagen verlinkt

- Hinzufügen von Metadaten der Publikation, die dann im E-Book gespeichert werden – die Publisher-Version des Creators bietet – wie oben erwähnt – vor allem publikationsrelevante Metadaten
- Hinzufügen eines so genannten Guides, einem Konzept, das die gezielte Führung durch ausgewählte Teile des E-Books ermöglichen soll

Die drei erstgenannten Ergänzungen gehören sinnvollerweise zu einer seriösen Produktion und sollten daher immer mit bedacht werden. Der letztgenannte Punkt des Guides ist nicht zwingend und vom Inhalt und von der Gliederung, letztlich vom Konzept des E-Books abhängig. In unserer Musterproduktion, die keine tiefe Gliederung besitzt, verzichten wir auf die Guide-Funktionalität, alle übrigen werden in der Folge berücksichtigt.

Fügen wir zunächst als ersten Schritt ein Cover-Bild hinzu, im folgenden Beispiel eine Grafikdatei cover . jpg, die in den Projektordner kopiert wird. Wählen wir dazu im links angezeigten View-Bereich den Eintrag »Cover Image« und fügen das Cover-Bild per Klick auf den Button »Add a cover image« hinzu (Abbildung 3.5).

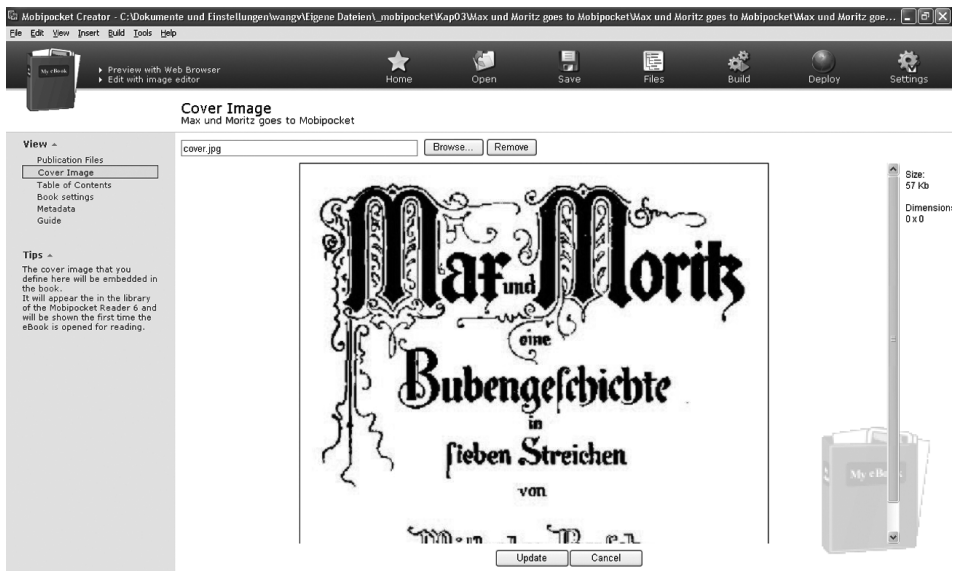
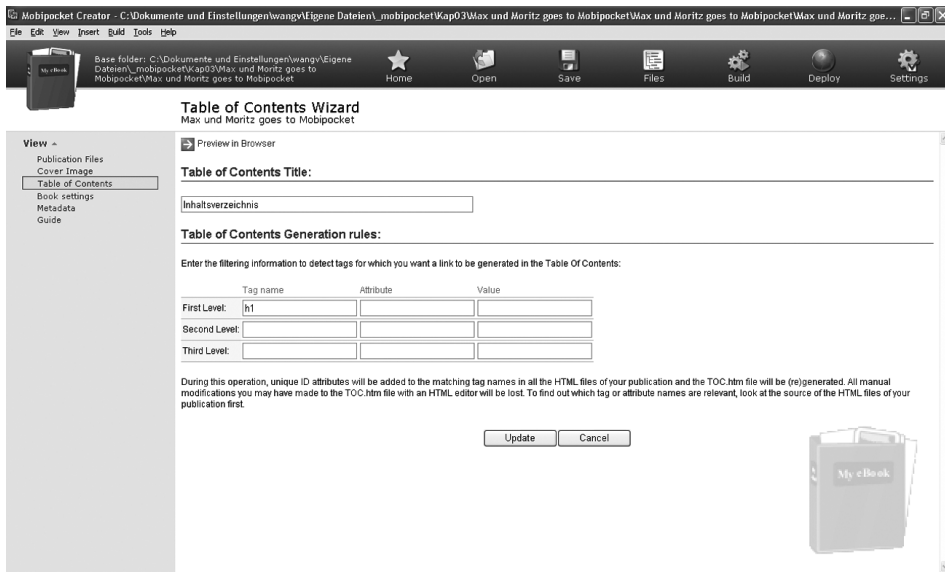


Abbildung 3.5: Hinzufügen der Cover-Grafik

Im zweiten Schritt ergänzen wir nun ein Inhaltsverzeichnis, das dem Benutzer die Navigation erleichtern soll. Dank der XHTML-Struktur des Quelldokuments können wir auf die Vorteile von XML aufsetzen und per Selektion der Überschriftenelemente das Inhaltsverzeichnis aufbauen. Wir

klicken hierfür auf den Eintrag »Table of Contents« und starten per Klick auf »Add a Table of Contents« den zugehörigen Assistenten. Als Titel tragen wir einfach »Inhaltsverzeichnis« ein und selektieren die Überschriftenebene des Quelldokuments, die den Kapiteln entspricht, nämlich das Element <h2> (<h1> wurde ja für den Titel des gesamten E-Books verwendet; siehe Abbildung 3.6).



**Abbildung 3.6:** Hinzufügen der Table-of-Contents-Datei

Das Ergebnis kann per Klick auf »Preview in Browser« angezeigt werden. Durch das abschließende Klicken auf »Update« wird die neue Datei des Inhaltsverzeichnisses den Publikationsdateien hinzugefügt.

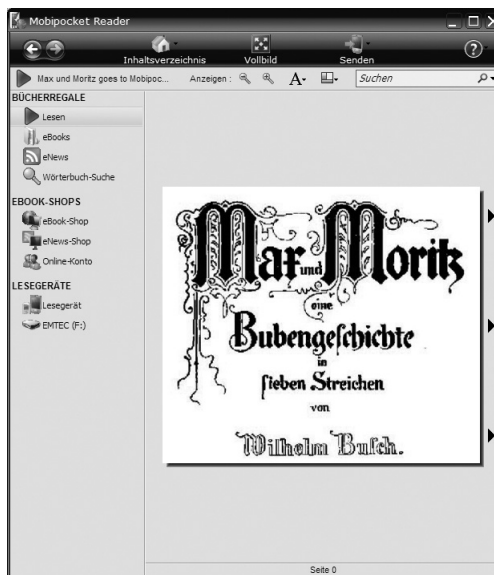
Schließlich tragen wir noch die fehlenden Metadaten der Publikation nach: Nach Wahl des Eintrages »Metadata« öffnet sich die Pflegemaske der publikationsrelevanten Metadaten, in die wir zunächst nur die wesentlichen Angaben einpflegen:

- Titel (Eingabe zwingend): »Max und Moritz goes Mobipocket«
- Autor: »Busch, Wilhelm«
- Verlag/Publisher
- Sprache (Eingabe zwingend): »German«
- Kategorie (Eingabe zwingend): »Classics«

- Beschreibung
- Review
- Publikationsdatum
- VorschauBild für die Online-Verkaufsdarstellung
- Demo PRC – falls eine spezielle Demofassung des E-Books gewünscht wird
- Verkaufspreis (Eingabe zwingend): »o«

Auch hier schließen wir die Eingabe durch Klick auf den »Update«-Button ab.

Nun sehen wir uns das Ergebnis der Verfeinerungen an. Nach einem erneuten Build-Lauf präsentiert sich unser Mobipocket-E-Book bereits wie folgt (Abbildung 3.7 bis Abbildung 3.9).



**Abbildung 3.7:** Ansicht des Cover-Bilds im Mobipocket Reader



Abbildung 3.8: Ansicht der Inhaltsübersicht

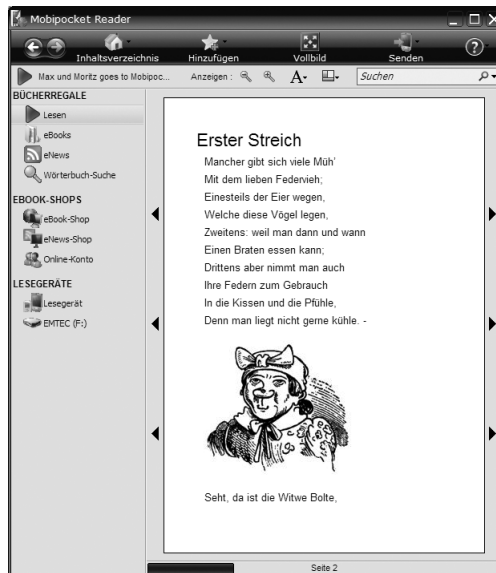


Abbildung 3.9: Ansicht nach Sprung aus dem Inhaltsverzeichnis in das erste Unterkapitel (dem ersten Streich)

### 3.3.4 Mobipocket im Detail

Nach diesem ersten Erfolgserlebnis lohnt sich ein Blick auf die hinter Mobipocket liegende Datenstruktur, die wir uns nun als Abschluss des vorliegenden Kapitels genauer ansehen wollen.

Der Projektordner enthält die folgenden Dateien (siehe Abbildung 3.10).

Name	Größe	Typ
Max und Moritz goes to Mobipocket_opfcache		Dateiordner
cover.jpg	58 KB	JPEG-Bild
Max und Moritz goes to Mobipocket.opf	4 KB	OPF-Datei
Max und Moritz goes to Mobipocket.prc	497 KB	ebook
Max_und_Moritz_001.jpg	18 KB	JPEG-Bild
Max_und_Moritz_002.jpg	12 KB	JPEG-Bild
Max_und_Moritz_003.jpg	23 KB	JPEG-Bild
Max_und_Moritz_004.jpg	24 KB	JPEG-Bild
Max_und_Moritz_005.jpg	32 KB	JPEG-Bild
Max_und_Moritz_006.jpg	29 KB	JPEG-Bild
Max_und_Moritz_007.jpg	23 KB	JPEG-Bild
Max_und_Moritz_008.jpg	30 KB	JPEG-Bild
Max_und_Moritz_009.jpg	24 KB	JPEG-Bild
Max_und_Moritz_010.jpg	29 KB	JPEG-Bild
Max_und_Moritz_011.jpg	23 KB	JPEG-Bild
Max_und_Moritz_012.jpg	22 KB	JPEG-Bild
Max_und_Moritz_013.jpg	28 KB	JPEG-Bild
Max_und_Moritz_014.jpg	33 KB	JPEG-Bild
Max_und_Moritz_015.jpg	39 KB	JPEG-Bild
Max_und_Moritz_016.jpg	37 KB	JPEG-Bild
max-und-moritz-streich1.html	5 KB	HTML-Datei
mbp_toc.html	2 KB	HTML-Datei

Abbildung 3.10: Inhalt des Projektordners des Mobipocket Creators

Die Datei Max und Moritz goes to Mobipocket.prc ist die kompilierte MOBI-Datei, also die fertige E-Book-Datei, die verteilt und vom Reader angezeigt werden kann. Alle JPG- und HTML-Dateien sind die Quelldateien, die dem Projekt zugrunde liegen und die als Publikationsdateien im Projektordner des Creators aufgeführt wurden.

Besonderes Interesse verdient nun die Datei Max und Moritz goes to Mobipocket.opf, die zentrale XML-Steuerdatei des Projektes. Wie bereits zu Beginn des Kapitels erwähnt, kennzeichnet die Endung .opf eine Datei der Open E-Book Publication Structure (OEBPS), die Sie bereits aus der Vorgeschichte von ePUB kennen (Abschnitt 2.1.1).

```
<?xml version="1.0" encoding="utf-8"?>
<package unique-identifier="uid">
  <metadata>
    <dc-metadata>
      xmlns:dc="http://purl.org/metadata/dublin_core"
      xmlns:oebpackage="http://openE-Book.org/namespaces/oeb-
      package/1.0/">
      <dc:Title>Max und Moritz goes to Mobipocket</dc:Title>
      <dc:Language>de</dc:Language>
      <dc:Identifier id="uid">42850E8D1F</dc:Identifier>
```

1  
2

```

<dc:Creator>Busch, Wilhelm</dc:Creator>
<dc:Subject BASICCode="FIC004000">Classics</dc:Subject>
</dc-metadata>
<x-metadata>
  <output encoding="Windows-1252"></output>
  <SRP Currency="EUR">0</SRP>
  <EmbeddedCover>cover.jpg</EmbeddedCover>
</x-metadata>
</metadata>
<manifest>
  <item id="item1" media-type="text/x-oebl-document"
    href="max-und-moritz-streich1.html"></item>
  <item id="item2" media-type="image/jpg"
    href="Max_und_Moritz_004.jpg"></item>
  <item id="item3" media-type="image/jpg"
    href="Max_und_Moritz_005.jpg"></item>
  <item id="item4" media-type="image/jpg"
    href="Max_und_Moritz_006.jpg"></item>
  <item id="item5" media-type="image/jpg"
    href="Max_und_Moritz_007.jpg"></item>
  <item id="item6" media-type="image/jpg"
    href="Max_und_Moritz_008.jpg"></item>
  <item id="item7" media-type="image/jpg"
    href="Max_und_Moritz_009.jpg"></item>
  <item id="item8" media-type="image/jpg"
    href="Max_und_Moritz_010.jpg"></item>
  <item id="item9" media-type="image/jpg"
    href="Max_und_Moritz_011.jpg"></item>
  <item id="item10" media-type="image/jpg"
    href="Max_und_Moritz_012.jpg"></item>
  <item id="item11" media-type="image/jpg"
    href="Max_und_Moritz_013.jpg"></item>
  <item id="item12" media-type="image/jpg"
    href="Max_und_Moritz_014.jpg"></item>
  <item id="item13" media-type="image/jpg"
    href="Max_und_Moritz_015.jpg"></item>
  <item id="item14" media-type="image/jpg"
    href="Max_und_Moritz_016.jpg"></item>
  <item id="item15" media-type="image/jpg"
    href="Max_und_Moritz_001.jpg"></item>
  <item id="item16" media-type="image/jpg"
    href="Max_und_Moritz_002.jpg"></item>
  <item id="item17" media-type="image/jpg"
    href="Max_und_Moritz_003.jpg"></item>
  <item id="toc" media-type="text/x-mbp-manifest-item">
    <process plugin-id="toc" id="mbp_toc_intermediary">

```

3

4

```

</level depth="1" tag="h2"></level>
</process>
<process plugin-id="apply-xsl" href="xmm:prev">
  <add-to-xsl
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="toc">
      <html>
        <head>
          <meta http-equiv="Content-Type"
            content="text/html; charset=utf-16"/>
        </head>
        <body>
          <h1 align="center">Inhaltsübersicht</h1>
          <ul>
            <xsl:for-each select="item">
              <xsl:call-template name="item"/>
            </xsl:for-each>
          </ul>
        </body>
      </html>
    </xsl:template>
    <xsl:template name="item">
      <li>
        <a href="{@href}">
          <xsl:value-of select="@title"/>
        </a>
      </li>
      <xsl:if test="item">
        <ul>
          <xsl:for-each select="item">
            <xsl:call-template name="item"/>
          </xsl:for-each>
        </ul>
      </xsl:if>
    </xsl:template>
  </add-to-xsl>
  <output href="mbp_toc.html"></output>
</process>
</item>
</manifest>
<spine>
  <itemref idref="toc"/><itemref idref="item1"/>
  <itemref idref="item2"/><itemref idref="item3"/>
  <itemref idref="item4"/><itemref idref="item5"/>
  <itemref idref="item6"/><itemref idref="item7"/>
  <itemref idref="item8"/><itemref idref="item9"/>

```

```

<itemref idref="item10"/><itemref idref="item11"/>
<itemref idref="item12"/><itemref idref="item13"/>
<itemref idref="item14"/><itemref idref="item15"/>
<itemref idref="item16"/><itemref idref="item17"/>
</spine>
<tours></tours>
<guide>
  <reference type="toc" title="Inhaltsübersicht"
    href="mbp_toc.html"></reference>
</guide>
</package>

```

⑥

⑦

**Listing 3.2:** Quellcode der Max-und-Moritz-OPF-Datei

Betrachten wir nun die wesentlichen Elemente in dieser XML-Datei:

❶ Das Wurzelement der Datei heißt `<package>` und beinhaltet alle für OPF relevanten Informationen.

❷ Es folgen die Metadaten `<metadata>`, die in zwei unterschiedliche Kategorien aufgeteilt sind: `<dc-metadata>` beinhaltet alle Metadaten, die im sogenannten Dublin-Core-Standard abbildbar sind. Diese Kategorie an Metadaten bringt einen eigenen Namensraum (`xmlns:dc="http://purl.org/metadata/dublin_core"`) mit. Alle nicht im Dublin-Core abbildbaren Metadaten werden in einem zweiten Metadaten-Element `<x-metadata>` hinzugefügt.

❸ Das Manifest-Element führt alle im E-Book beinhalteten Quelldateien auf, pro Datei wird ein `<item>`-Element geschrieben. Hier finden wir also zunächst die XHTML-Datei `max-und-moritz-streich1.html` (`id="item1"`), dann sämtliche JPG-Grafiken (`id="item2"` bis `id="item17"`) und schließlich auch die vom Creator erzeugte Inhaltsverzeichnisdatei (`id="toc"`).

❹ Es folgen zwei `<process>`-Elemente, die der Creator für die Generierung der Inhaltsverzeichnisdatei benötigt. Diese Elemente sind nicht Bestandteil der offiziellen OEBPS-Standards, also proprietäre Erweiterungen des Mobipocket-Formates.

❺ Das Element `<spine>` (wörtlich das »Rückgrat«, »Buchrücken«) legt die Abfolge der Inhalte innerhalb des E-Books fest. Da die Inhalte bereits oben innerhalb des `<manifest>`-Elements festgelegt wurden, genügt nun die Referenzierung der zugehörigen `id`-Attribute.

❻ und ❼: Die beiden Elemente `<tours>` und `<guide>` sind nur der Vollständigkeit halber mit aufgeführt; da sie in der Musterproduktion von Mo-

bipocket keine Funktion haben, gehe ich auf sie an dieser Stelle nicht näher ein.

Damit schließen wir dieses Kapitel ab. Zusammengefasst kann festgehalten werden, dass das Mobipocket-Format ein auf XML basierendes E-Book-Format darstellt, das auf den XML-Standards XHTML und OEBPS beruht. Der Mobipocket-Creator zeigt exemplarisch, dass es durchaus Werkzeuge zur Erstellung von E-Books gibt, die eine rasche E-Book-Produktion auch ohne tiefe technische Kenntnis ermöglichen. Allerdings gelingt dies nur dann, wenn eine gute Datengrundlage vorliegt. Das Beispiel Mobipocket weist auch die grundlegenden Konzepte auf, die einem modernen E-Book-Format zugrunde liegen. Der zuletzt genannte Aufbau der OEBPS-Datei wird uns auch im ePUB-Format wieder begegnen.

### Anmerkungen

- 1 Siehe <http://www.mobipocket.com/dev/>.
- 2 Siehe <http://www.mobipocket.com/en/DownloadSoft/application.asp?device=Others>; Unterstützte Systeme: Blackberry, Windows Mobile, Symbian OS, Palm OS sowie diverse E-Book-Reader: CyBook, iLiad, Hanlin, BE-Book (Stand September 2010).
- 3 Die folgenden Angaben stützen sich hauptsächlich auf Informationen der offiziellen Entwicklerseite von Mobipocket (<http://www.mobipocket.com/dev/article.asp?BaseFolder=prcgen&File=mobiformat.htm>) sowie auf <http://wiki.mobileread.com/wiki/MOBI>.
- 4 Hierzu detaillierter <http://www.mobipocket.com/dev/article.asp?BaseFolder=prcgen&File=building.htm>.
- 5 Die Texte sind auch im Gutenberg-Projekt zu finden, siehe <http://www.gutenberg.org/E-Books/17161>.