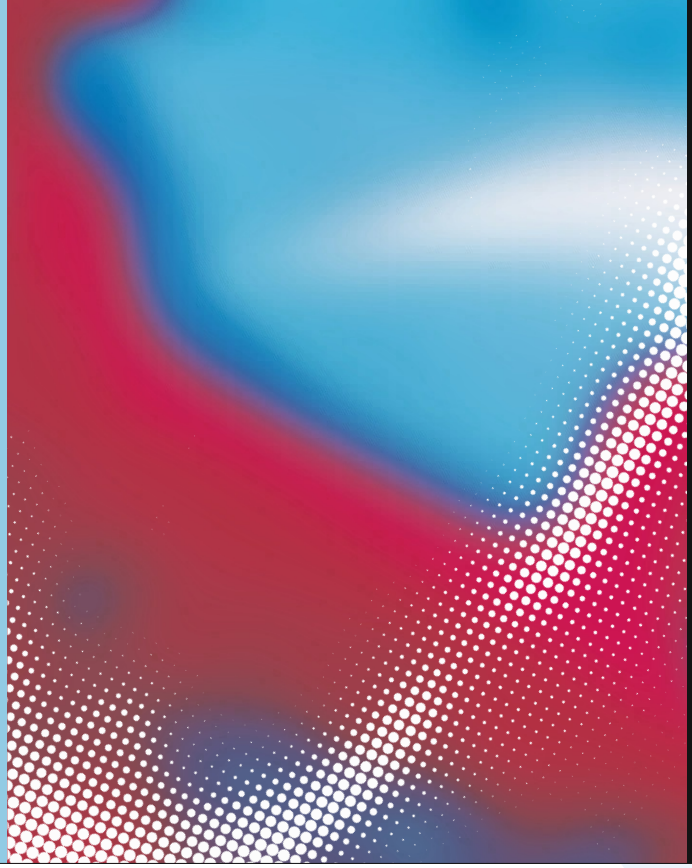
X.media.press



Thomas Hoffmann-Walbeck et al.

X.media.press ist eine praxisorientierte Reihe zur Gestaltung und Produktion von Multimedia-Projekten sowie von Digital- und Printmedien.

Standards in der Medienproduktion

Springer Vieweg


X.media.press

X.media.press ist eine praxisorientierte Reihe zur Gestaltung und
Produktion von Multimedia-Projekten sowie von Digital- und Printmedien.

Thomas Hoffmann-Walbeck •
Gottfried Zimmermann • Marko Hedler •
Jan-Peter Homann • Alexander Henka •
Sebastian Riegel • Ansgar Gerlicher •
Martin Goik • Christophe Strobbe

Standards in der Medienproduktion

Springer Vieweg

Prof. Dr. Thomas Hoffmann-Walbeck
Hochschule der Medien
Stuttgart, Deutschland

Prof. Dr.-Ing. Marko Hedler
Hochschule der Medien
Stuttgart, Deutschland

Alexander Henka
Hochschule der Medien
Stuttgart, Deutschland

Prof. Dr. Ansgar Gerlicher
Hochschule der Medien
Stuttgart, Deutschland

Christophe Strobbe
Hochschule der Medien
Stuttgart, Deutschland

Prof. Dr. Gottfried Zimmermann
Hochschule der Medien
Stuttgart, Deutschland

Jan-Peter Homann
homann colormanagement
Berlin, Deutschland

Sebastian Riegel
Hochschule der Medien
Stuttgart, Deutschland

Prof. Dr. Martin Goik
Hochschule der Medien
Stuttgart, Deutschland

ISBN 978-3-642-15042-5 ISBN 978-3-642-15043-2 (eBook)
DOI 10.1007/978-3-642-15043-2

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

Springer Vieweg

Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier.

Springer Vieweg ist eine Marke von Springer DE. Springer DE ist Teil der Fachverlagsgruppe Springer Science+Business Media
www.springer-vieweg.de

Abkürzungsverzeichnis

µm	Mikrometer
AFP	Advanced Function Presentation
AMF	Apache Mobile Filter
AM	Amplitudenmoduliert
AMS	American Mathematical Society
ANSI	American National Standards Institute
APP	Advanced Print Publisher
APPE	Adobe PDF Print Engine
ASCII	American Standard Code Information Interchange
AV	Automatische Vererbung von einem äußeren auf ein inneres Element
BDE	Betriebsdatenerfassung
BMP	Basic Multilingual Plane
BVDM	Bundesverband Druck und Medien
CAD	Computer-Aided Design
CC/PP	Composite Capabilities / Preferences Profile
CCITT	Comité Consultatif International Téléphonique et Télégraphique
CFF	Compact Font Format
CFF2	Common File Format, Version 2
CGATS	Committee for Graphic Arts Technologies Standards
CIP3	International Cooperation for Integration of Prepress, Press and Postpress
CIP4	International Cooperation for Integration of Processes in Prepress, Press and Postpress
CIPA	Camera & Imaging Products Association
CMYK	Cyan Magenta Yellow Black
CPSI	Configurable PostScript Interpreter
CSS	Cascading Style Sheets
CSV	Comma Separated Values
CTAN	Comprehensive TeX Archive Network
CTM	Current Transformation Matrix
CtP	Computer-to-Plate
CWD	Current Working Directory
DAM	Digital Asset Management
DANTE	Deutschsprachige Anwendervereinigung Tex e.V.
DCMI	Dublin Core Metadata Initiative
DCS	Desktop Color Separation
DDR	Device Description Repositories
DFE	Digital Front End
DNG	Digital Negative
DPI	Dots per Inch
DTD	Document Type Definition
DTP	Desktop Publishing

DXF	Drawing Interchange File Format
EBNF	Erweiterte Backus-Naur-Form
ECI	European Color Initiative
ECMA	European Computer Manufacturers Association
EMF	Windows Enhanced Metafile
EOT	Embedded Open Type
EPS	Encapsulated PostScript
EPUB	Electronic Publication
ERP	Enterprise Resource Planning
Exif	Exchangeable Image File Format
FM	Frequenzmoduliert
FOGRA	Forschungsgesellschaft Druck e.V.
FTP	File Transfer Protocol
GB	Gray Box
GCR	Grey Component Replacement
GDI	Graphical Device Interface
GIF	Graphics Interchange Format
GPS	Global Positioning System
GWG	Ghent PDF Workgroup
HDTV	High Definition Television
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICC	International Color Consortium
ICS	Interoperability Conformance Specification
IDPF	International Digital Publishing Forum
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IFD	Image File Directory
IFRA	World Association of Newspapers and News Publishers
IGES	Initial Graphics Exchange Specification
IIM	Information Interchange Model
IPDS	Intelligent Printer Data Stream
IPTC	International Press Telecommunications Council
ISBN	International Standard Book Number
ISO	International Organization for Standardization
ISSN	International Standard Serial Number
ITU	International Telecommunication Union
JBIG	Joint Bi-level Image Experts Group
JDF	Job Definition Format
JEITA	Japan Electronics and Information Technology Industries Association
JFIF	JPEG File Interchange Format
JMF	Job Messaging Format
JPG	Dateieendung für JFIF- oder SPIFF-Dateien
JPEG/JPG	Joint Photographic Experts Group
JSP	Java Server Pages
Lab	(Farbcharakterisierung)
LCD	Liquid Crystal Display
LZW	Lempel, Ziv und Welch
MIME	Multipurpose Internet Mail Extensions
MIS	Management Information System
MO:DCA	Mixed Object Document Content Architecture
MP3	MPEG-1 Audio Layer III oder MPEG-2 Audio Layer III

NAA	Newspaper Association of America
OCF	Open Container Format
OEM	Original Equipment Manufacturer
OMA	Open Mobile Alliance
OPI	Open Prepress Interface
OTF	OpenType Format
PCL	Printer Command Language
PDF	Portable Document Format
PDF/VT	PDF Variable/Transactional
PDF/X	PDF Exchange
PDL	Page Description Language
PFR	Portable Font Resource
PJTF	Portable Jobticket Format
PNG	Portable Network Graphics
PODi	Print On Demand Initiative
PPD	PostScript Printer Description
PPF	Print Production Format
PPI	Pixel per Inch
PPML	Personalized Print Markup Language
PS	PostScript
PSO	ProzessStandard Offsetdruck
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RFC	Request for Comments
RGB	Rot Grün Blau
RIP	Raster Image Processor
RLE	Run Length Encoding
SGML	Standard Generalized Markup Language
SMIL	Synchronized Multimedia Integration Language
SPARQL	Protocol And RDF Query Language
SPIFF	Still Picture Interchange File Format
SVG	Scalable Vector Graphics
SVGA	Super Video Graphics Array
TEI	Text Encoding Initiative
TIFF	Tag(ged) Image File Format
TIFF/IT	TIFF/Image Technology
TTF	TrueType Format
UAProf	User Agent Profile
UCR	Under Color Removal
UCS	Universal Multiple-Octet Coded Character Set
UCS	Universal Character Set
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF	UCS Transformation Format
VDP	Variable-Data Printing
VIPP	Variable-Data Intelligent PostScript Printware
VPS	Variable Print Specification
W3C	World Wide Web Consortium
WALL	Wireless Abstraction Library
WML	Wireless Markup Language
WMS	Workflow Management System
WOFF	Web Open Font Format
WPF	Windows Presentation Foundation

WURFL	Wireless Universal Resource File Language
WWW	World Wide Web
WYSIWYG	What You See Is What You Get
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XMP	Extensible Metadata Platform
XPS	XML Paper Specification
XSD	XML Schema Definition
XSLT	Extensible Stylesheet Language Transformation
YCC	(Farbcharakterisierung)
ZIP	Zipper (englisch Reißverschluss – ein Datenformat)

Inhaltsverzeichnis

Grundlagen

1

Marko Hedler, Thomas Hoffmann-Walbeck, Jan-Peter Homann

In diesem Kapitel werden die Grundlagen der Medienvorstufe vorgestellt. Nach einer Einführung in die allgemeine Thematik und dem Überblick zu diesem Buch in Abschn. 1.1 wird in den folgenden Abschnitten die technologische Basis gelegt. Zunächst wird das Thema PDF behandelt. Hierbei geht es um die Funktionen der unterschiedlichen PDF-Versionen, um die Erstellung von PDF-Dateien und auch um die internen Strukturen des Datenformats. Im folgenden Abschn. 1.3 werden dann die grundlegenden Eigenschaften von Farben und Farbmodellen dargestellt. Der Abschn. 1.4 ist mit „Einführung strukturierter Dokumente“ überschrieben, worin auch ein kurzer Exkurs in die sogenannten „Auszeichnungssprachen“ gegeben wird. Mit dem Thema „Metadaten“ wird das Kapitel abgeschlossen. Dabei werden abstrakte Merkmale solcher Daten diskutiert, aber auch bereits einige konkrete Beispiele präsentiert.

1.1 Einführung und Übersicht

Seit einigen Jahren wird der Begriff „Medienkonvergenz“, also das Zusammenwachsen der Medien, diskutiert. Das betrifft Medieninhalte, wirtschaftliche Verflechtungen der Medienproduzenten und auch die technischen Produktionsverfahren. Letzteres wird mit dem Begriff „Cross Media Publishing“ noch genauer gefasst. Unter einer solchen „medienübergreifenden Veröffentlichung“ wird vor allem das parallele Publizieren auf unterschiedlichen Kanälen, wie Print, Online und mobilen Medien verstanden (*Multichannel Publishing*). Grundvoraussetzung hierzu ist die medienneutrale Datenhaltung. Die Daten müssen dabei so angelegt sein, dass sie als Quelle für die Ausgabe in unterschiedlichen Medien geeignet sind.

Die Produktionsverfahren für die unterschiedlichen Medien sind aber vielfach immer noch sehr disparat. Das Herstellen der Daten eines Druckproduktes in der Druckvorstufe hat beispielsweise zunächst einmal nur wenig gemein mit der Programmierung einer App für ein Smartphone. Und es ist beileibe (noch) nicht so, dass aus einem Datenbestand quasi auf Knopfdruck mal eine Webseite, mal eine gedruckte Broschüre oder aber eine Anwendung für ein iPad entsteht. Die Produktionswerkzeuge sind häufig sehr unterschiedlich, auch wenn sie vielleicht alle die gleichen Texte, Bilder und Grafiken (bzw. entsprechende Werkzeuge zur Erstellung dieser Ressourcen) verwenden. Trotzdem ist eine angehende Verschmelzung dieser Medienproduktionen zu erkennen – so werden zum Beispiel viele E-Books mit den gleichen Methoden hergestellt, die auch zur Herstellung von klassischen Büchern benutzt werden. Dieser Trend wird sicher weiter fortschreiten, schon aus wirtschaftlichen Zwängen. Denn kleinere und mittelgroße Unternehmen beispielsweise scheuen den großen Aufwand mannigfacher Medienproduktionen, die zurzeit nötig sind, um Werbekampagnen auf unterschiedlichen Kanälen gleichzeitig zu führen.

Inhalte müssen auf den verschiedenen Ausgabekanälen unterschiedlich dargestellt werden, schon allein wegen der sehr verschiedenen Ausgabegrößen. Eine Überschrift in einem gedruckten Produkt hat typischerweise einen anderen Schriftgrad als auf einem Smartphone, wo er außerdem auch noch variabel sein sollte. Damit ist eine Trennung von Inhalt, Struktur und Layout für das Cross Media Publishing eine zwingende Voraussetzung, was auch mit dem Schlagwort „strukturierte Dokumente“ bezeichnet wird.

Obwohl die Applikationen zur Produktion von Medieninhalten auf unterschiedlichen Kanälen verschieden sein mögen, haben sich in den letzten Jahren mehr und mehr die folgenden drei gemeinsamen Grundlagen herausgebildet:

- Das *Portable Document Format* (PDF)
- Die *Extensible Markup Language* (XML) bzw. die verwandte *Hypertext Markup Language* (HTML)
- Metadatenformate zur Beschreibung der Dokumente und teilweise auch zur Festlegung der Medienproduktionen

Durch diese Datenformate und Sprachen, die von unterschiedlichen Mediendienstleistern eingesetzt werden, sind nicht nur elementare technische Gemeinsamkeiten vorgegeben, sondern auch indirekt die Verwendung von gleichen Software-Tools bei der Produktion.

T. Hoffmann-Walbeck et al., *Standards in der Medienproduktion*, X.media.press,
DOI 10.1007/978-3-642-15043-2_1, © Springer-Verlag Berlin Heidelberg 2013

Der Einsatz von gleichen Datenformaten für sehr unterschiedliche Medien wirft aber auch einige Probleme auf. So dürfen beispielsweise interaktive Elemente in einer PDF-Datei, die für eine Online-Veröffentlichung sinnvoll sind, bei einer Printproduktion nicht vorkommen. Dieser Umstand macht weitere Spezifikationen erforderlich, die bestimmen, wie ein und dasselbe Datenformat für die unterschiedlichen Ausgabekanäle aussehen sollte. Ein weiterer wichtiger Aspekt in diesem Zusammenhang stellt die Definition von Farben dar, die sich ebenfalls stark zwischen den verschiedenen Medien unterscheiden.

In diesem Buch wird auf die oben genannten Datenformate und Sprachen eingegangen sowie auf die Produktionsmethoden zur Herstellung digitaler Publikationen unterschiedlicher Arten. Die Autoren orientieren sich dabei an Standards – sei es von der *International Organization for Standardization* (ISO), dem *World Wide Web Consortium* (W3C), Industriestandards oder De-facto-Standards von Herstellern und Verbänden. Es werden zwar teilweise auch die Tiefen der Datenstrukturen und Produktionsmethoden behandelt, dennoch können hier nur Einführungen in die verschiedenen Technologien gegeben werden: Für ein vertieftes Studium wird auf entsprechende Originalquellen oder auch spezielle Lehrbücher verwiesen.

Das Buch ist in vier Kapitel untergliedert.

1. Einführung

In diesem Kapitel werden zunächst die technischen Grundlagen zu der bekannten Seitenbeschreibungssprache PDF und Farbdefinitionen von beliebigen, digitalen Objekten behandelt. In dem anschließenden Abschnitt wird erklärt, warum eine Dokumentenstrukturierung für die medienneutrale Datenhaltung notwendig ist. Insbesondere wird auch beschrieben, wie man mittels einer Auszeichnungssprache Dokumente strukturieren kann. Im letzten Abschnitt dieses Kapitels geht es allgemein um Metadaten, also um Informationen, die zwar nicht unmittelbar zum Anzeigen oder auch zum Drucken von Inhalten nötig sind, aber einen großen Stellenwert in der Medienproduktion einnehmen, da sie unter anderem Automatisierungsmöglichkeiten in der Medienproduktion bieten.

2. Content- und Metadaten

Im zweiten Kapitel werden die Grundbausteine näher beschrieben, aus denen sich digitale Publikationen zusammensetzen: Texte, Grafiken und Bilder.

In Abschn. 2.1 wird als Erstes die Zeichencodierung ausgeführt, also letztendlich wie Buchstaben, Ziffern und Interpunktionszeichen in Bits und Bytes repräsentiert werden können. Mit Hilfe der Zeichencodes entstehen Textformate wie *CSV*, *Rich Text Format* (RTF) oder *TeX/LaTeX*. Besondere Aufmerksamkeit verdient aber in diesem Abschnitt die *Extensible Markup Language* (XML), ein textbasiertes Format, mit dem beliebige, digitale Dokumente definiert werden können.

Der Abschn. 2.2 ist der Vektorgrafik (= Objektgrafik) gewidmet, also mathematischen Beschreibungen von Linien und Kurven. Schriftzeichen werden zur Darstellung meist über solche Vektorgrafik definiert, was Thema von Abschn. 2.3 ist. In Abschn. 2.4 geht es um den Aufbau digitaler Bilder, also Strukturen, die durch einzelne Farbpixel beschrieben werden.

Das Thema „Farbe“ betrifft alle diese „Grundbausteine“ und wird deswegen separat in Abschn. 2.5 vertieft. Schließlich werden in Abschn. 2.6 die wichtigsten Metadatenformate vorgestellt, die beim elektronischen Publizieren und bei der Herstellung von Printprodukten Verwendung finden.

3. Standards im elektronischen Publizieren

Das dritte Kapitel widmet sich den Standards für das elektronische Publizieren. Die *HyperText Markup Language* (HTML 5) und die *Cascading Style Sheets* (CSS) bilden für verschiedenste Anwendungen und Ausgabegeräte eine wesentliche Basis, wobei Webseiten das wichtigste Teilsegment darstellen. Diese Technologien werden in Abschn. 3.1 demonstriert.

Eine Einführung in das EPUB-Format für elektronische Bücher findet man in Abschn. 3.2. Es zeigt in der Version 3 viele Verwandtschaften mit dem zuvor diskutierten HTML5.

Beim Umformen von XML-Dateien für verschiedene Anwendungsbereiche oder der Wandlung in unterschiedliche Ausgabeformate wie zum Beispiel PDF sind XSLT, XPath und XSL-FO von zentraler Bedeutung. Diese Technologien werden in Abschn. 3.3 behandelt.

Sollen Dokumente in den Formaten HTML5, EPUB oder PDF barrierefrei zugänglich sein, so gibt es hierfür jeweils eigene Vorgaben, die in Abschn. 3.5 vorgestellt werden.

Ein weiterer Ausgabekanal für Dokumente sind mobile Endgeräte. Auf mobile Anwendungen, welche mittels Webtechnologien erstellt werden, sogenannte Web-Apps, wird in Abschn. 3.6 eingegangen. Es wird beschrieben, mit welchen Verfahren und Technologien das Problem der Diversität der Endgeräte angegangen wird. Ein Schlagwort ist dabei „Responsive Webdesign“, das in Abschn. 3.6.2 dargestellt wird. In Abschn. 3.6.3. werden verschiedene Frameworks zur Web-App-Entwicklung behandelt. Eine Besonderheit stellen die sogenannten hybriden Apps dar, welche die Vorteile einer Web-App und einer nativen App vereinen sollen. Diese werden in Abschn. 3.6.4 beschrieben.

4. Standardisierte Druckproduktion

Für die Druckproduktion gibt es sowohl für die Herstellung als auch für die Kontrolle von Druckdaten eine Reihe von ISO-Standards, auf die in Abschn. 4.1 eingegangen wird. In Abschn. 4.2 wird der PDF/X-Standard erläutert, der auf PDF basiert und alle für den Druck notwendigen Elemente festlegt sowie die potentiell störenden PDF-Elemente aus-

schließt. Anschließend werden in Abschn. 4.3 die Möglichkeiten zur Kontrolle und Korrektur von PDF-Daten vorgestellt. Eine Druckerei wird schließlich diese PDF-Daten weiter aufbereiten, sie über-/unterfüllen (*Trapping*), mehrere in geeigneter Weise auf einen Druckbogen platzieren (*Bogenmontage*) und sie schließlich für die Ausgabe konvertieren (*Ripping*). Zur Kontrolle wird zuerst die Ausgabe meist auf einem Tintenstrahldrucker oder auf einem Monitor vorgenommen (*Formproof*) und erst danach auf einem *Computer-to-Plate*-Belichter. Alle diese Vorstufenprozesse werden in Abschn. 4.4 beschrieben. Da das Drucken variabler Daten aus einer Datenbank heraus besondere Anforderungen stellt, werden diese in einem eigenen Unterabschnitt veranschaulicht.

Sollen Abläufe in der Druckerei automatisiert werden, so wird meist das Metadatenformat *Job Definition Format* (JDF) verwendet, auf das in Abschn. 4.5 im Detail eingegangen wird.

Das Buch endet bei der Datenaufbereitung. Die eigentliche Druckformherstellung für die unterschiedlichen Druckverfahren, also die Herstellung von Druckplatten, Tiefdruckzylinder oder Siebe für den Siebdruck, werden hier nicht mehr behandelt. Das Gleiche gilt für die Druckprozesse sowie für alle Arbeitsschritte in der Druckweiterverarbeitung.

Dem Springer-Verlag möchten wir ganz herzlich für die Unterstützung und Geduld danken – insbesondere Frau Dorothea Glaunsinger, Frau Gabriele Fischer und Herrn Hermann Engesser. Unser besonderer Dank gilt aber der Lektorin, Frau Ursula Zimpfer, die durch ihren Sachverstand und ihre Genauigkeit ganz wesentlich zur Qualität des Buches beigetragen hat.

1.2 Einführung in PDF

1985 wurde die Seitenbeschreibungssprache PostScript (PS) von der Firma Adobe Systems veröffentlicht (Adobe 1999b). Damals gab es mehrere alternative Sprachen dieser Art, wobei sich *PostScript* durchgesetzt hat. Bei allen war die geräteunabhängige Beschreibung von Druckdaten das wichtigste Designziel, das heißt, die Positionen und Größenangaben der grafischen Objekte (Bilder, Schriftzeichen, Vektorgrafik) wurden nicht mehr in Pixel für ein gewisses Gerät mit einer speziellen Auflösung angegeben, sondern in Inch, Punkt oder Zentimeter. Erst das Endgerät – oder genauer gesagt der *Raster Image Processor* (RIP) des Endgerätes – hat dann diese abstrakten Weltkoordinaten in konkrete Gerätekoordinaten umgerechnet. Das PostScript selbst war dabei für den Benutzer transparent: Das heißt, ein Druckertreiber hat im Hintergrund PostScript erzeugt, das dann zu einem Drucker oder einem Belichter geschickt wurde.

PostScript ist eine interpretative Programmiersprache. PS-Befehle „zeichnen“ auf einem (virtuellen) leeren Blatt Papier ein opakes, grafisches Objekt nach dem anderen. Die Interpretation der PS-Daten auf einem Endgerät sollte dann eigentlich auch in der gleichen Reihenfolge erfolgen wie die ursprüngliche Definition der grafischen Objekte. Natürlich können so auch mehrere Seiten hintereinander in eine PS-Datei geschrieben werden, dann aber sollten auch die Seiten entsprechend nacheinander interpretiert werden.

Das ist in der Praxis natürlich nicht einzuhalten und schon sehr bald wurden zusätzliche Konventionen definiert, die es ermöglichen sollten, einzelne Seiten unabhängig voneinander zu interpretieren (*Document Structuring Conventions* (Adobe 1996a)). Doch auch weiterhin blieb PostScript eher ein Ausgabe- und nicht so sehr ein Austauschformat. Erst die Weiterentwicklung zum *Portable Document Format* (PDF) im Jahr 1993 änderte diese Situation. Allerdings war das neue Format erst 1996 mit der Version 1.2, aber vor allem 1999 mit der Version 1.3 für die Druckvorstufe geeignet.

Das PDF ist in gewisser Weise das Nachfolgerformat von PostScript. Insbesondere ist das grafische Modell von PDF dem von PostScript ähnlich, wenn es auch in den letzten Jahren viele Erweiterungen erfahren hat. PostScript selbst ist nur noch recht mittelbar im Gebrauch, sei es als *Encapsulated PostScript* (Adobe 1992b), sei es als Zwischenformat zur Generierung von PDF oder intern in RIPs. Und natürlich lässt sich PDF immer in PostScript wandeln (durch einen PS-Druckertreiber) und umgekehrt (durch den *Acrobat Distiller* oder ein vergleichbares Programm).

Mittlerweile hat PDF einen ungeheuren Umfang an Möglichkeiten, und es gibt eine Menge Literatur über das Thema, beispielsweise (Schneeberger 2008), (Adobe 2006) oder (Merz und Drümmer 2002). In diesem Kapitel werden deswegen auch nur ein paar grundsätzliche Eigenschaften vom PDF vorgestellt. Einige vertiefende Dinge findet man in anderen Abschnitten dieses Buches (wie Schriftformate in Abschn. 2.3, Objektgrafiken in Abschn. 2.2, Bildformate in Abschn. 2.4, PDF/X in Abschn. 4.2 und barrierefreies PDF in Abschn. 3.5), andere werden hier gar nicht aufgeführt. Insbesondere werden in diesem Abschnitt die interaktiven Merkmale von PDF weniger behandelt.

1.2.1 Überblick zu den Versionen

Die Abb. 1.1 zeigt die unterschiedlichen Versionen von PostScript- und PDF-Spezifikationen und wann sie publiziert wurden. Neue PDF-Versionen gingen immer auch mit neuen Versionen des *Adobe Acrobats* einher. Hier werden kurz die herausragenden Merkmale der einzelnen PDF-Versionen vorgestellt, soweit sie auch für den Druckbereich von Belang sind.

Erst mit der Version 1.2 hatte PDF einen ausreichenden Umfang von Eigenschaften, um für die Druckvorstufe interessant zu sein. Die Version 1.0 kannte beispielsweise nur den RGB-Farbraum. Mit der Version 1.2 wurde auch der CMYK-

Versionen von PostScript und PDF

1985	PS Level 1
1986	PS RIP für Linotype
1991	PS Level 2
1993	PDF 1.0 / Acrobat 1.0
1994	PDF 1.1 / Acrobat 2.0
1996	PDF 1.2 / Acrobat 3.0
1997	PostScript 3
1999	PDF 1.3 / Acrobat 4.0
2001	PDF 1.4 / Acrobat 5.0
2003	PDF 1.5 / Acrobat 6.0
2005	PDF 1.6 / Acrobat 7.0
2007	PDF 1.7 / Acrobat 8.0 (ISO 32000-1)

Abb. 1.1 PostScript- und PDF-Versionen

Farbraum unterstützt. Ab dann gab es auch die Möglichkeit, Sonderfarben und die Überdrucken-Eigenschaft in PDF zu definieren. Trotzdem gab es noch ein paar Einschränkungen, die erst mit der Version 1.3 überwunden wurden.

So wurde in PDF 1.3 der *DeviceN*-Farbraum eingeführt, womit man beispielsweise einen *Hexachrome*-Farbraum[1] oder auch Bilder im *Duplexdruck*[2] (auch *Duotone*- oder allgemeiner *Mehrkanalbilder* genannt) mit Sonderfarben unterstützen kann.

Auch der Endformatrahmen (*Trim Box*) und der Beschnittrahmen (*Bleed Box*), die heute unverzichtbar erscheinen (s. Abschn. 1.2.3), wurden erst bei dieser Version hinzugefügt.

Schließlich sollte noch das Objekt *Shading Pattern* zur Definition von weichen Verläufen erwähnt werden. Bis dahin waren PDF-Verläufe durch Flächen mit gewissen Farbwertabstufungen realisiert worden. Diese Technik war auch bei PostScript Level 1 und Level 2 üblich, hatte aber den Nachteil, dass Verläufe nicht immer in der bestmöglichen Auflösung des Ausgabegerätes erzeugt wurden und deswegen streifig aussahen. Doch bereits in PostScript 3 kann man Verläufe zunächst in einer abstrakten Notation unabhängig von der Auflösung des Ausgabegerätes festlegen. Dann entscheidet erst das RIP des Ausgabegerätes, wie fein die Verläufe tatsächlich abgestuft werden. Mit der Version 1.3 hat dieses qualitativ bessere Verfahren bezüglich Verläufen auch bei PDF Einzug gehalten.

Mit der Version 1.4 wurde das klassische PostScript/PDF-Modell erweitert. Denn ab dieser Version sind nicht mehr alle Objekte opak, sondern können auch als transparent definiert werden. Das hatte allerdings im Druckbereich viele Probleme zur Folge. In den RIPs wurden (und werden teilweise immer noch) intern die einkommende PDF-Dateien zu PS konvertiert. PostScript kennt aber keine transparenten Objekte, so dass diese folglich erst einmal flach gerechnet werden müssen (siehe weiter unten den Absatz über „Überducken, Aussparen, Transparenzen“). Da das aber nicht immer fehlerfrei gelingt, werden so in den RIPs gelegentlich mangelhafte Daten produziert, die im Druck für unangenehme Überraschungen sorgen können.

Außerdem wurde das *Tagged PDF* eingeführt. Damit können den eigentlichen Inhaltsdaten zusätzliche strukturelle Informationen hinzugefügt werden, wodurch es möglich wird, beispielsweise für Sehbehinderte Hinweise bezüglich einer Sprachausgabe anzuhängen, so dass dann nicht mehr formale Dinge wie Seitenzahlen oder Kopf- und Fußleisten vorgelesen werden. In dem Abschnitt „Tagged PDF“ unten wird dieses Thema behandelt.

Des Weiteren wurde die JBIG2-Kompression für monochrome Bilder in die Spezifikation aufgenommen. Damit lassen sich solche Binärbilder sehr gut entweder verlustfrei oder aber auch (noch effektiver) verlustbehaftet komprimieren. Das Verfahren wird in Abschn. 2.4 ausgeführt.

Mit PDF 1.4 wurde auch die Struktur *Output Intent* eingeführt. Damit kann die Druckbedingung – also Farbmodell und Ausgabegerät – charakterisiert werden, für die das PDF generiert wurde. Eine Druckbedingung wird dabei üblicherweise durch einen Namen eines ICC-Profils oder durch ein eingebettetes ICC-Profil selber definiert (vgl. Abschn. 2.5.4).

Als letzte Neuerung in PDF 1.4 soll noch das Referenzieren auf andere PDF-Dokumente erwähnt werden. Damit können in einem Dokument X auf einzelne Seiten (oder Objekte) eines anderen Dokumentes Y verwiesen werden. Dafür ist in der Datei X dann entweder nur Name und Pfad der PDF-Datei Y eingetragen oder die referenzierte Seite von Y ist direkt als ein Objekt in der Datei X enthalten. Letzteres ist ein Beispiel für das *Embedded File Stream*-Konzept, das seit der Version 1.3 Teil der PDF-Spezifikation ist.

Die PDF-Version 1.5 enthält auch noch einen Filter zur Decodierung von Bildern, die mit der *JPEG 2000*-Methode komprimiert worden sind. Diese sowohl verlustfreie als auch verlustbehaftete Komprimierungsmethode (ISO/IEC 2004b) hat im Vergleich zum regulären *JPEG* (ISO/IEC 1994) einige Vorteile, insbesondere aber eine bessere Kompressionsrate. Details findet man in Abschn. 2.4.

Mit der Version 1.5 können auch PDF-Ebenen (*Optional Content*) gesetzt werden, das heißt, Grafikelemente werden in Ebenen zusammengefasst. Diese sind dann mit einer geeigneten Applikation einzeln ein- und ausblendbar. Damit lassen sich nicht nur Sprachversionen voneinander trennen, sondern auch die CAD-Zeichnungen von den Druckdaten bei der Faltschachtelherstellung oder auch bei Multilayer-Karten verschiedenen Kartenebene. Die Ebenen werden in Layoutprogrammen erstellt und dann bei der PDF-Konvertierung

[1] Hierbei werden 6 Farben verwendet: Cyan, Magenta, Gelb, Schwarz, Grün und Orange.
[2] Ein Graubild mit einer weiteren Farbe.

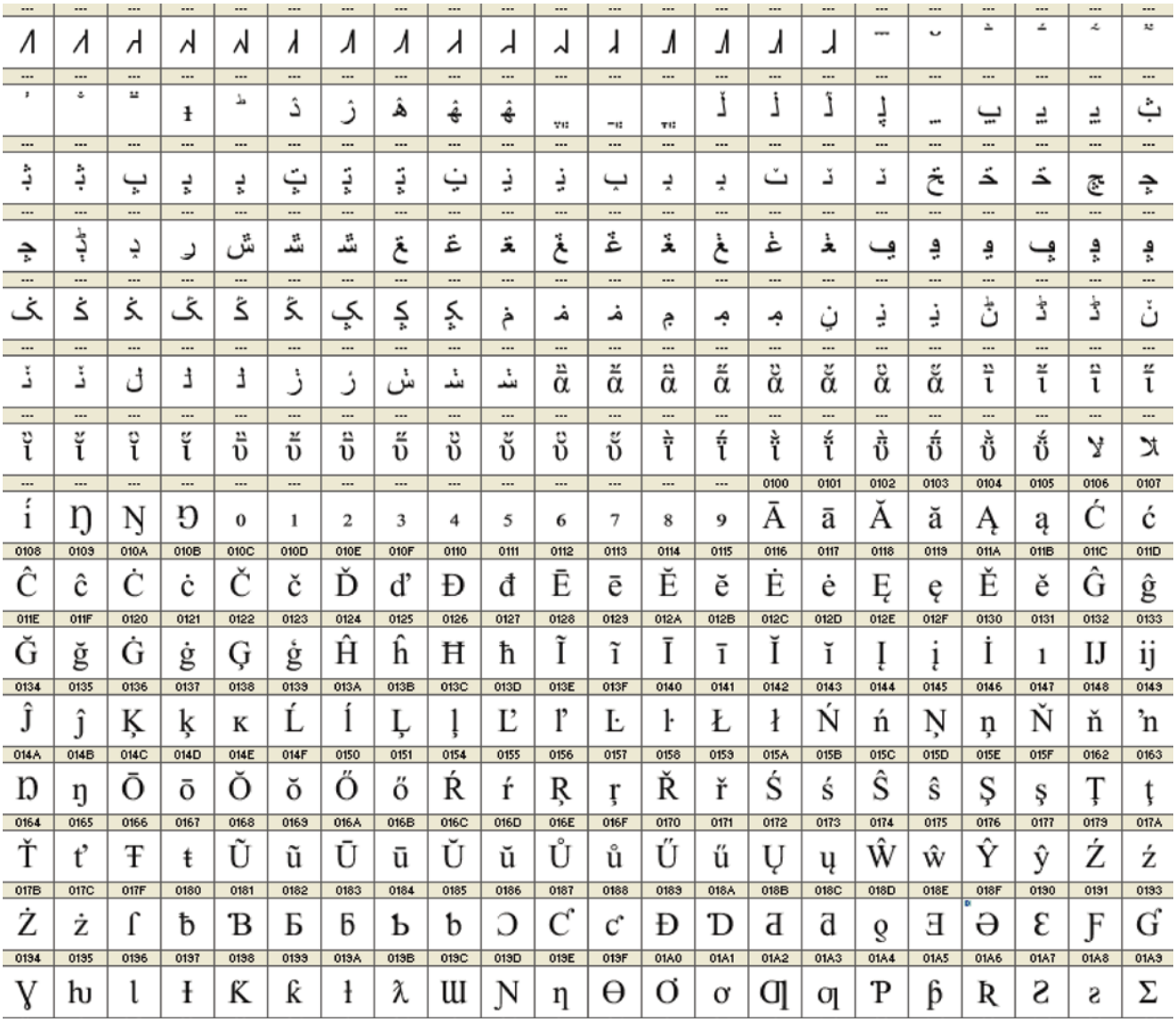


Abb. 1.2 Teil des Zeichensatzes der Schrift Times-New-Roman

übernommen. Alternativ kann auch mit geeigneter Software aus mehreren PDF-Dateien eine neue mit mehreren Ebenen zusammengestellt werden.

Mit dieser Version wurden auch 12-Bit und 16-Bit-Bilder eingeführt. Die maximale Bittiefe[3] pro Farbauszug (*Bits Per Component*) wurde für PDF damit von 8 auf 16 erhöht.

Seit der Version 1.6 ist die Einbettung von *OpenType*-Schriften möglich. Vereinfacht gesagt, ist das *OpenType*-Format ein Zusammenschluss von dem klassischen PostScript-*Type-1*-Schriftformat von Adobe und dem *TrueType*-Format von Apple. *Type 1*-Fonts gibt es bereits seit 1984, also seit der Einführung von PostScript. Die *TrueType*-Technologie wurde hingegen erst 1991 von Apple veröffentlicht und dann ein Jahr später auch an Microsoft lizenziert. 1996 wurde dann das *OpenType*-Format von Microsoft und Adobe herausgegeben. Die Schriftzeichen einer *OpenType*-Schrift werden dabei entweder in der Art von *Type 1* oder von *TrueType* beschrieben. Für den Anwender sind jedoch vor allem zwei Dinge bei diesem Schriftentyp von Bedeutung: Erstens sind *OpenType*-Schriften plattformübergreifend hinsichtlich der beiden Betriebssysteme von Apple und Microsoft, zweitens können sie (anders zumindest als *Type-1*-Schriften) sehr viele Zeichen enthalten, so dass nicht nur viele Schriftsysteme (kyrillische, arabische, fernöstliche, …) unterstützt werden können, sondern auch erweiterte typografische Funktionen (s. Abschn. 2.3). Abbildung 1.2 zeigt einen kleinen Ausschnitt des (mehrere Tausend Zeichen enthaltenden) Zeichensatzes der *OpenType*-Schrift *Times-New-Roman* von Windows 7. Vor der PDF-Version 1.6 musste eine *OpenType*-Schrift immer in *Type 1* oder in *TrueType* konvertiert werden, bevor sie

[3] Auch Farbtiefe genannt (s. Abschn. 2.4).

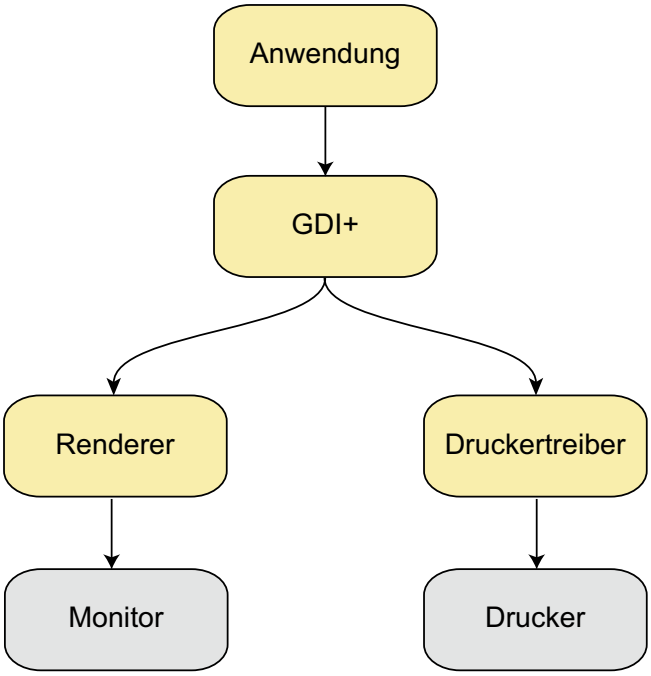


Abb. 1.3 Betriebssysteminterne Grafikbefehle werden durch einen Druckertreiber in eine Seitenbeschreibungssprache konvertiert

in PDF eingebettet werden konnte. Mit der Version 1.6 können *OpenType*-Fonts jedoch als solche eingebettet werden. Für Details über Schriftformate s. Abschn. 2.3.4.

Die maximale PDF-Seitengröße betrug bis zur Version 1.6 genau 5,08 × 5,08 Meter. Das hat nicht immer ausgereicht. Doch mittlerweile kann die Länge oder Breite einer Seite bis zu beachtlichen 381 Kilometer groß sein.

PDF-Version 1.7 ist die Basis für die ISO-Norm 32000, die 2008 publiziert wurde (ISO 2008b). Danach hat Adobe Systems keine neueren Versionen von PDF veröffentlicht, allerdings regelmäßig Zusätze zur Version 1.7 publiziert, beispielsweise (Adobe 2008) und (Adobe 2009a).

Grundsätzlich sollte man sich bei der PDF-Generierung auf niedere Versionen beschränken und nicht unbedingt die allerneueste Version herstellen, die von der eigenen Software unterstützt wird. Denn dann besteht immer die Gefahr, dass nachfolgende Programme, die das PDF importieren und verarbeiten müssen, mit neuen PDF-Konstrukten nicht zurechtkommen. Seit Langem waren in der Druckindustrie die PDF-Versionen 1.3 und 1.4 üblich. Doch mittlerweile wird in der Regel eher Version 1.6 empfohlen. Das Thema wird in Abschn. 4.2 über PDF/X vertieft.

1.2.2 PDF-Erstellung

PDF kann auf zwei sehr unterschiedliche Arten generiert werden:

- mit PostScript als Zwischenformat und
- ohne PostScript als Zwischenformat.

Im ersten Fall wird beispielsweise mit einem PS-Druckertreiber zunächst PostScript erzeugt und dieses dann in PDF konvertiert. In letzteren Fall wird das PDF häufig mittels einer PDF-Bibliothek direkt von der Applikation heraus generiert. Man kann sich leicht vorstellen, dass die PDF-Dateien in beiden Fällen sehr unterschiedlich sein können.

Zunächst soll die Situation untersucht werden, in der ein PS-Druckertreiber verwendet wird. Die Abb. 1.3 zeigt das Prinzip am Beispiel der klassischen Windows-Umgebung. Applikationen dürfen nicht selbst die Pixel des Monitors ansteuern, sondern müssen stattdessen eine Schnittstelle zum Betriebssystem bedienen, damit dieses alle grafischen Objekte am Bildschirm anzeigt. Diese Schnittstelle hat beim klassischen Windows den Namen *Graphical Device Interface* (GDI) bzw. die verbesserte Version heißt GDI+. Damit kann die Applikation grafische Inhalte mit geräteunabhängigen Funktionen einer Grafikbibliothek beschreiben. Diese wiederum rufen dann spezifische Funktionen eines gewählten Gerätetreibers auf. Somit muss sich ein Anwendungsentwickler nicht um die Details des Ausgabegerätes (wie Bildschirm oder Drucker) kümmern. Außerdem impliziert diese Softwarearchitektur, dass aus einem Datensatz heraus Bildschirm und Drucker ihre Informationen bekommen (WYSIWYG-Prinzip).

Die Folge von Grafikbefehlen für die GDI+-Schnittstelle können auch in einem eigenen Datensatz gespeichert werden, der dann *Windows Enhanced Metafile* (EMF) oder auch EMF+ heißt. Viele einfache Drucker (sogenannte *GDI-Drucker*) erhalten ein vom GDI bereits fertig aufbereitetes Bild, das nur noch ausgedruckt werden muss. Doch professionellere Drucker werden mit unterschiedlichen Seitenbeschreibungssprachen (englisch: *Page Description Language* oder PDL) angesteuert. So versteht zum Beispiel ein PS-Drucker PostScript und ein PCL-Drucker die von Hewlett-Packard entwickelte Sprache *Printer Command Language* (PCL). Das EMF+ muss folglich noch in eine Seitenbeschreibungssprache konvertiert werden. Genau das machen Druckertreiber, die mit dem Betriebssystem mitgeliefert werden. Ein solcher PS-Druckertreiber konvertiert also universell EMF+ in PostScript.

Anstelle von GDI+ werden auch andere Schnittstellen verwendet wie *Quarz* in OS X oder *Windows Presentation Foundation* (WPF), das ab 2007 mit *Windows Vista* ausgeliefert wird. WPF beruht auf einer eigenen Dokumentenbeschreibungssprache, nämlich auf *XML Paper Specification* (XPS), die im letzten Teil dieses Abschnitts vorgestellt wird. Die beiden Druckpfade GDI+ und XPS, die in neueren Windows-Systemen aus Kompatibilitätsgründen gleichzeitig enthalten sind, werden in Abb. 1.4 skizziert. Spezielle Übergänge lassen den Wechsel von GDI+ auf XPS und umgekehrt zu. Der praktische Unterschied für den Anwender zwischen einer GDI+-Anwendung und einer WPF-Anwendung sind die Druckdialogmasken. Während nämlich beim GDI+ sowohl

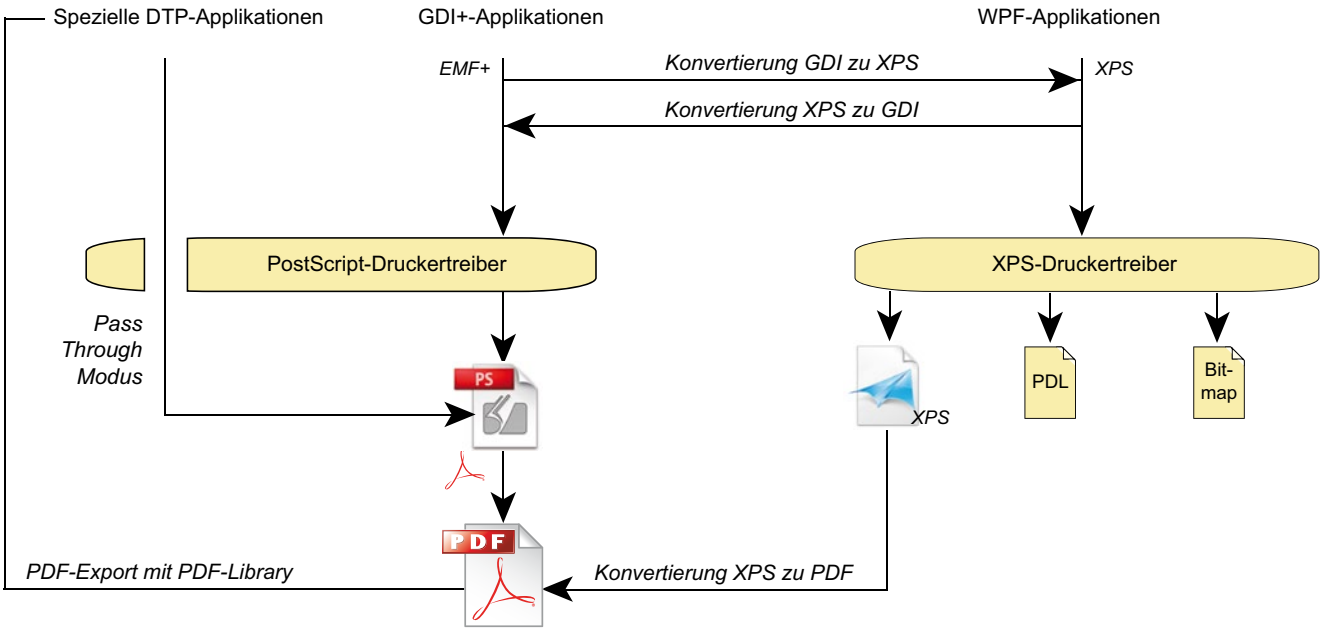


Abb. 1.4 In neueren Windows-Versionen wird zusätzlich zu GDI intern auch das XPS-Format verwendet

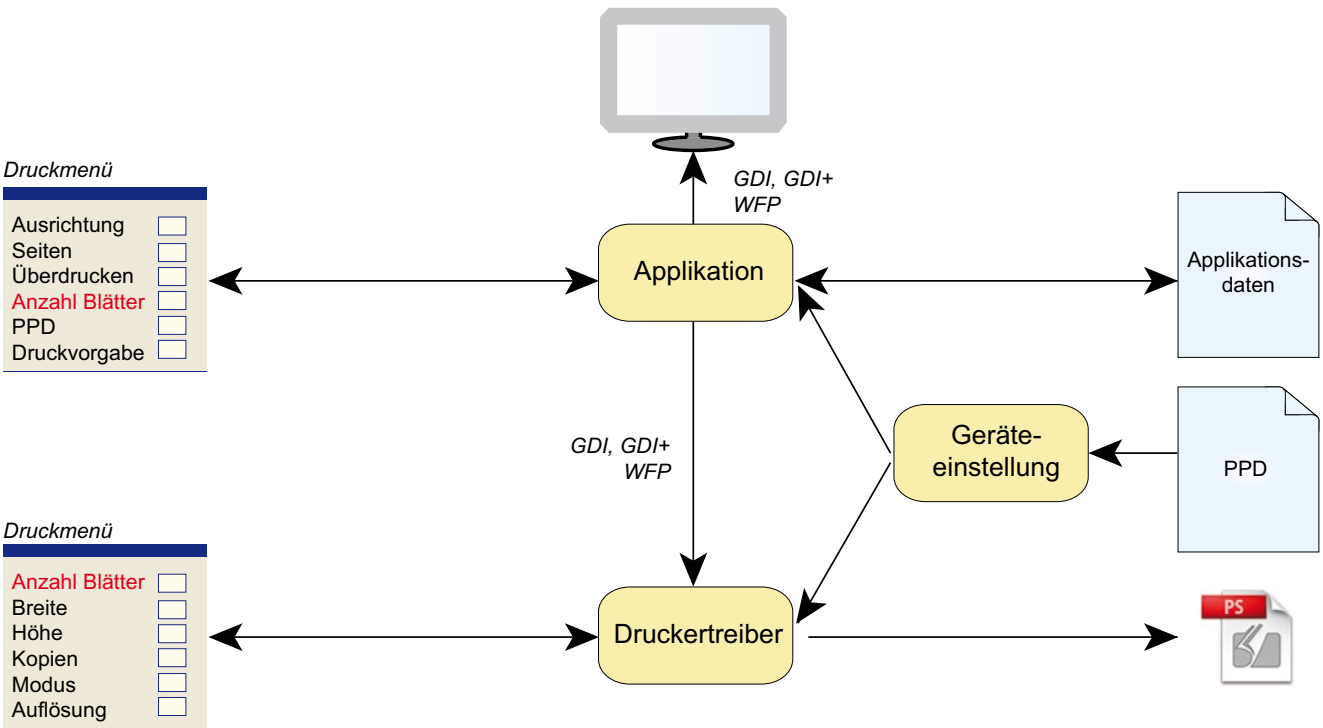


Abb. 1.5 Unter GDI werden sowohl von der Applikation als auch vom Druckertreiber Dialogmasken für den Druck generiert. Hierfür werden Informationen aus der PPD und den Geräteeinstellungen verwendet

die Applikation als auch der Druckertreiber Dialogmasken für den Benutzer bereitstellen, ist das bei WPF mit Hilfe von sogenannten *Print Tickets* integriert. Damit werden die Anwender nicht mehr durch doppelte Abfragen zu Druckangaben verwirrt. Denn bei GDI+-Dialogen kann es beispielsweise passieren, dass sowohl die Applikation als auch der Druckertreiber die Information „Seiten pro Blatt“ einzeln abfragt (s. Abb. 1.5). Wenn man dann beispielsweise in beiden Masken „zwei Seiten“ angibt, dann werden vier Seiten auf einem Blatt gedruckt.

In der Abb. 1.5 erkennt man außerdem die Rolle von der *PostScript Printer Description* (PPD) und der *Geräteeinstellung*. Eine PPD-Datei (Adobe 1996b), (Adobe 1997) ist eine Textdatei, die den Leistungsumfang eines PostScript-

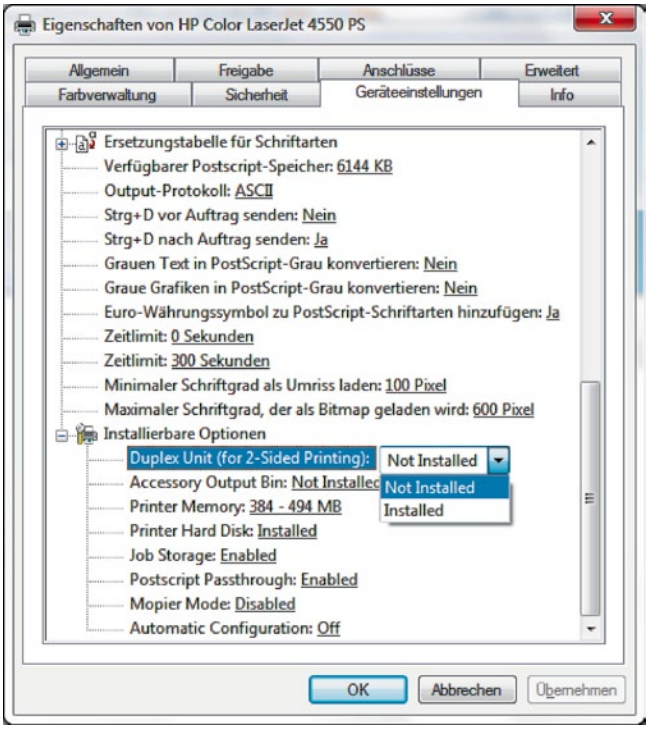


Abb. 1.6 Beispiel einer Geräteeinstellung unter Windows

Druckertyps beschreibt. Dabei werden sehr unterschiedliche Dinge definiert wie Papiergrößen, die verarbeitet werden können, Ausbaustufen des Arbeitsspeichers, im RIP des Gerätes enthaltene Schriften, Duplexeinheiten[4] und Zusatzeinrichtungen zum Klammern. Einige dieser Eigenschaften sind dabei optional, wie beispielsweise zusätzliche Papierschächte eines Druckers oder auch die Arbeitsspeichergröße. Diese können dann unter Windows in den Geräteeinstellungen noch dem tatsächlichen Ausbau eines konkreten Druckers angepasst werden. Auch die Default-Einstellungen bezüglich Farbe, Papier und dergleichen können hier getroffen werden. Die Abb. 1.6 zeigt eine solche Geräteeinstellung unter Windows 7. Die PPD wird zusammen mit der Geräteeinstellung dann auch dazu verwendet, die Druckdialoge sowohl der Applikation als auch die des Druckertreibers auf den Drucker anzupassen. So werden beispielsweise nur die Papiergrößen angeboten, die der Drucker verarbeiten kann, und auch die Vorauswahl auf das Standardpapierformat wird hierüber gesteuert. Man bemerke, dass eine PPD mit dem Namen *Adobe PDF* häufig zur Herstellung von PDF über PostScript herangezogen wird.

Eine Applikation kann also mit der GDI+-Schnittstelle und einem generischen Druckertreiber PostScript erzeugen, das seinerseits von einem geeigneten Programm wieder zu PDF konvertiert wird. Es ist klar, dass dann Zwischenformate wie EMF+, XPS, PostScript oder auch der PS-Druckertreiber Auswirkung auf das generierte PDF haben. Ganz besonders deutlich wird das bei den transparenten Objekten, wie in Abschn. 1.2.3 im Absatz „Überdrucken, Aussparen, Transparenzen“ dargelegt wird.

Um den Einfluss des Druckertreibers bei der PostScript- bzw. PDF-Erstellung auszuschalten, kann eine Applikation ihn auch zu einem gewissen Grade umgehen und selbst die Seitenbeschreibungssprache erzeugen. Dieser sogenannte *Pass-Through-Modus* des PS-Druckertreibers wird vor allem von professioneller DTP-Software verwendet. Die Datenflüsse sind in der schematischen Abb. 1.4 eingezeichnet.

Die direkte Erzeugung von PDF-Dateien durch die Applikation (also ohne den Zwischenschritt PostScript) hat jedoch in der letzten Zeit in der grafischen Industrie immer mehr an Bedeutung gewonnen. Anfänglich war diese Verfahrensweise durch sporadisch auftretende Fehler in Verruf geraten. Das lag wohl einerseits an den Applikationen bzw. an den eingesetzten Softwarebibliotheken selbst, die bei der Konvertierung schlicht fehlerhaftes PDF generiert haben, andererseits auch an der Tatsache, dass viele RIPs immer noch aus dem eingehenden PDF zunächst PostScript erzeugen, um Letzteres zu verarbeiten. So kann die frühe Konvertierung in PostScript mit einer anschließenden Konvertierung in PDF die Wahrscheinlichkeit der korrekten Verarbeitbarkeit der PDF-Datei im RIP erhöhen. Mittlerweile wurden aber die meisten Softwarebibliotheken zur PDF-Erstellung verbessert und viele Grafik- und Layoutprogramme erzeugen beim direkten PDF-Export durchaus korrektes PDF.

In der Praxis werden häufig fehlerhafte PDF-Dateien erzeugt. Vor allem bei der PDF-Generierung über PostScript müssen zwei Sätze von Dialogmasken komplett richtig ausgefüllt werden, nämlich die zur PostScript-Generierung und die zur anschließenden Konvertierung von PostScript zu PDF. Welche typischen Fehler hierbei gemacht werden, wird in den Abschn. 4.2 und 4.3 diskutiert.

Die derzeit gültige Empfehlung ist, PDF über das Druckmenü zu erzeugen, wenn entweder die Applikation keinen direkten Export einer PDF-Datei zulässt oder wenn der Druckdienstleister ein auf PostScript basierendes RIP verwendet. Adobe und seine Lizenznehmer bezeichnen diesen RIP-Typ mit *Configurable PostScript Interpreter* (CPSI). Diese RIP-Architektur wurde von der sogenannten *Adobe PDF Print Engine* (APPE) abgelöst, die PDF nativ verarbeiten kann. Wird ein solches RIP verwendet, ist ein direktes Erzeugen der PDF-Datei vorzuziehen.

1.2.3 PDF-Seitenrahmen

Schon bei PostScript gab es für viele grafische Strukturen die *BoundingBox* (*BBox*), ein Rechteck minimaler Größe, das ein grafisches (und beliebig geformtes) Objekt umschließt. Natürlich kann man dort auch eine Papiergröße (*PageSize*) definieren, auf die ausgedruckt wird.

[4] Vorrichtungen zum Rückseitendruck.

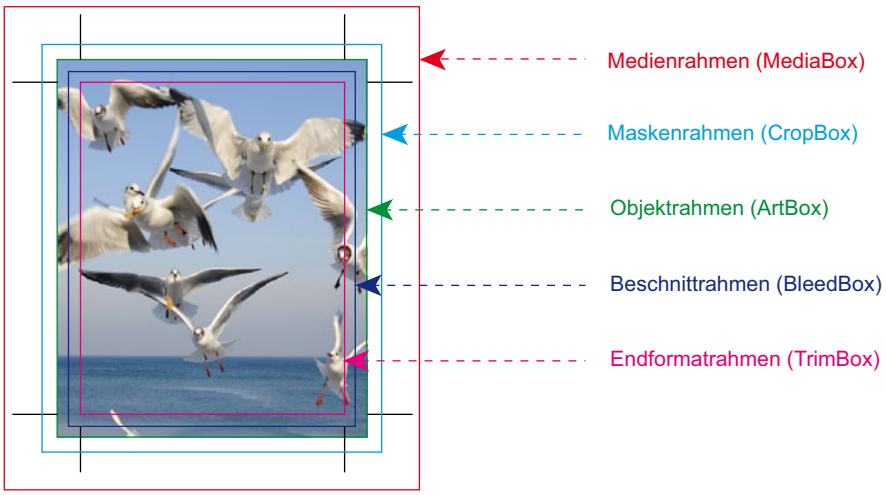


Abb. 1.7 PDF-Rahmen

In PDF unterscheidet man mehrere *Boxen*, die im Deutschen auch „Rahmen“ genannt werden. Diese sind vor allem für die automatische Positionierung der Seiten bei der Montage von großer Wichtigkeit. Die Abb. 1.7 zeigt die verschiedenen Rahmen im Überblick. Die Werte werden standardmäßig in 1/72 Inch angegeben.

- Der Medienrahmen (*MediaBox*) gibt die Größe des Papiers oder eines anderen Mediums an, auf dem die PDF-Datei ausgedruckt oder angezeigt werden soll.
- Ein Maskenrahmen (*CropBox*) definiert ein Rechteck, das alle Elemente auf der Seite umfasst, die sichtbar sein sollen. Außerhalb dieser *CropBox* werden alle Objekte weggeschnitten.
- Der Anschnittrahmen oder Beschnittrahmen (*BleedBox*) bezeichnet die Größe des Endformates eines Druckproduktes zuzüglich des Beschnitts. Ein Beschnitt (von typischerweise drei Millimeter) wird angelegt, damit nicht Ungenauigkeiten beim Falzen oder beim Schneiden dazu führen, dass am Produktrand feine Linien in der Farbe des Bedruckstoffs zu sehen sind. Ein Beschnitt sollte immer dann angelegt werden, wenn grafische Objekte bis zum Rand des Endformates heranreichen. Im Layout muss dann ein solches Objekt über das Endformat hinausreichen.
- Der Endformatrahmen (*TrimBox*) bezeichnet die Größe des Endformates eines Druckproduktes, also die Größe, auf die durch einen 3-Seiten-Beschnitt das Endprodukt zugeschnitten wird.
- Objektrahmen (*ArtBox*) kann ein beliebiger Ausschnitt einer Seite sein, den der Gestalter als den beabsichtigten Darstellungsbereich angibt. Dieser Rahmen kann dazu verwendet werden, um das PDF-Dokument in einer anderen Applikation zu positionieren.

Die Größen der drei wichtigsten Rahmen werden durch die folgende Beziehung definiert:

$$\text{Endformatrahmen} \leq \text{Anschnittrahmen} \leq \text{Medienrahmen}$$

Zu beachten ist, dass diese Rahmenangaben bis auf den Medienrahmen nur optional sind. Für PDF/X-Dateien ist es jedoch erforderlich, dass auch der Endformatrahmen angegeben ist (s. Abschn. 4.2).

Überdrucken, Aussparen, Transparenzen

Im PDF (wie auch bei PostScript) decken standardmäßig die mit Farbe gefüllten Objekte alle darunter liegenden Objekte ab. Auch DTP-Programme sind grundsätzlich so voreingestellt. Das bedeutet, dass ein Objekt in beliebiger Farbe durchaus jede andere eingefärbte Fläche komplett abdeckt. Natürlich wäre das bei den körperlichen Druckfarben nicht der Fall, denn die sind in der Regel lasierend. Insofern muss in der Vorstufe der Hintergrund „ausgespart“ werden. In Abb. 1.8 ist in der obersten Zeile ein magentafarbener Stern konstruiert, der den Cyan-Hintergrund gewissermaßen ausstanzt. Im Englischen wird dieser Vorgang auch treffend *knock-out* genannt. Dass dem so ist, kann man gut an den beiden Farbauszügen von Cyan und Magenta erkennen.

Aber sowohl mit PostScript als auch PDF kann man grafische Objekte auch auf überdrucken stellen. Dann werden alle Farben hinter einem Objekt beibehalten, so dass sich im Druck eine Mischfarbe ergibt, wie in Zeile 2 zu erkennen ist. In einigen Layoutprogrammen wird die Überdrucken-Eigenschaft im Druck auch in der Ansicht optional simuliert (Überdruckenvorschau).

Zeile 3 der Abbildung zeigt die Situation, wenn der Stern auf 50 % transparent eingestellt ist. In diesem Fall wird sowohl der Cyan-Hintergrund als auch der Magenta-Stern selbst mit einem Tonwert[5] von 50 % gedruckt. Stellt man das Magenta-Objekt auf 60 % transparent, also 40 % deckend ein, so wird der Tonwert von Cyan auf 60 % gesetzt und der von dem Magenta-Objekt auf 40 %. Wird jedoch das Cyan-Rechteck

[5] Ein Tonwert beschreibt Helligkeitsstufen einer Farbe und wird in 0 % bis 100 % (= Vollton) gemessen. Der Begriff „Farbwert“ wird manchmal synonym verwendet.

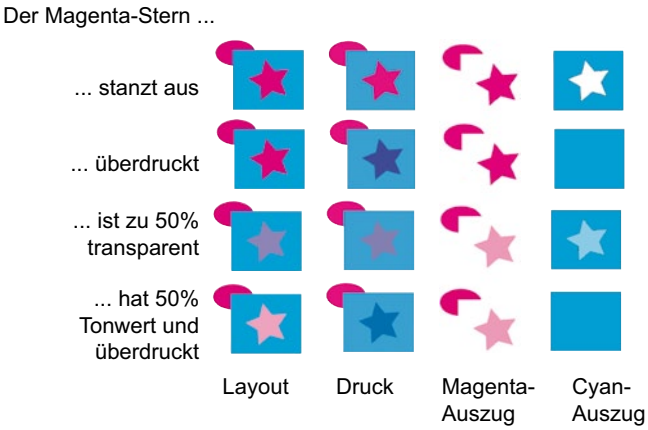


Abb. 1.8 Defaultmäßig stanzen oben liegende Objekte die unteren aus. Man kann sie aber auch auf „überdrucken“ stellen oder eine individuelle Deckkraft (Transparenz) festlegen

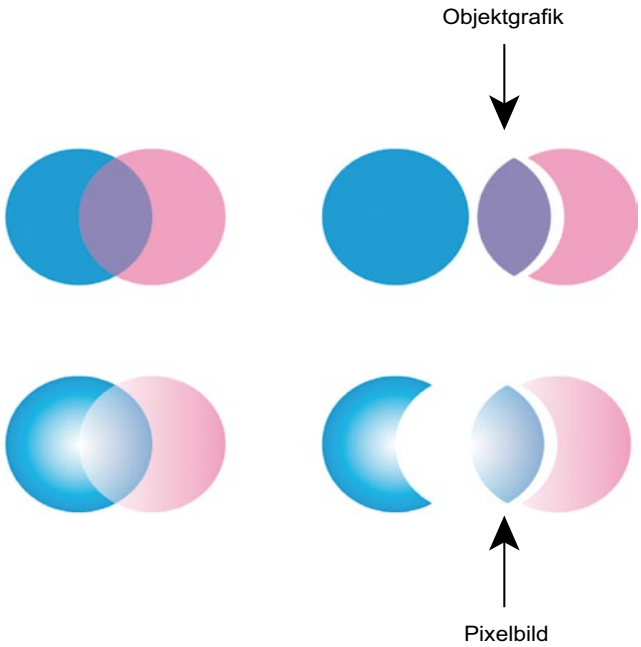


Abb. 1.9 Wenn transparente Objekte flachgerechnet (reduziert) werden, entstehen neue Objektgrafiken oder Bildelemente

nicht im Vollton, sondern mit einem Tonwert von 70 % eingefärbt und darauf ein Magenta-Objekt mit 40 % Deckkraft gelegt, so erhält der Cyan-Hintergrund nur noch 70 % von dem 60%igen Tonwert, also genau 42 %.

Einstellungen zur Deckkraft einer Farbe sind in Grafik- und Layoutprogrammen schon seit Langem möglich. Da jedoch weder in PostScript noch in PDF vor der Version 1.4 ein solches Grafikattribut definierbar war, mussten in diesem Fall die einzelnen übereinanderliegenden Schichten flachgerechnet werden. Was das bedeutet, kann man in Abb. 1.9 in der ersten Zeile erkennen. Im Layoutprogramm wurden hier zwei Kreise definiert, wobei der oben liegende, magentafarbene Kreis auf 50 % transparent gestellt wurde. Bei der Umwandlung zu PostScript oder PDF 1.3 entsteht dann ein neues Objekt in Form einer konvexen Linse. Letzteres ist in der Farbe eingefärbt, die sich durch das transparente Objekt im Vordergrund und dem Hintergrund ergibt. Zur Verdeutlichung wurden die drei entstandenen PDF-Objekte etwas auseinandergezogen. Dabei ist das neu generierte, linsenförmige Objekt auf „aussparend“ gesetzt. Die Stellen vom Cyan-Kreis, die hinter dem magentafarbenen Kreis liegen, werden also letztendlich eliminiert. Das geschieht aber erst im RIP, im PDF bleibt der Cyan-Kreis noch erhalten.

Ab der PDF-Version 1.4 können PDF-Objekte nicht nur mit der Eigenschaft „aussparen“ und „überdrucken“ versehen werden, sondern auch mit „transparent“. Dann können im PDF auch in diesem Fall die beiden Kreise vollständig erhalten bleiben, was eine nachträgliche Modifikation der PDF-Daten erleichtert.

Der Vorgang, Transparenzen flach zu rechnen, wird mit *Transparenzreduzierung* oder im Englischen mit *Flattening* bezeichnet. Doch nicht immer ist eine Transparenzreduzierung möglich, bei der eine oder mehrere neue Objektgrafik(en) entstehen. Es kann sein, dass ein Objekt generiert werden muss, das zu komplex ist, um es durch eine mit einer Farbe gefüllten Grafik zu beschreiben. Stattdessen wird dann ein

Pixelbild (oder auch mehrere Pixelbilder) generiert. In Zeile 2 von Abb. 1.9 ist das der Fall. Denn hier überlagern sich in transparenter Weise ein kreisförmiger Verlauf in dem linken Kreis und ein linearer Verlauf in dem rechten Kreis. Dort, wo beide sich treffen, wird ein Bild generiert, weil sich dieser neue Verlauf nicht mehr so einfach in mathematischer Weise beschreiben lässt.

Um die Unterschiede zur Transparenz noch einmal zu verdeutlichen, zeigt die Zeile 4 von Abb. 1.8 die Situation, in der ein Stern mit 50 % Tonwert in Magenta die Cyan-Fläche überdruckt. In allen vier Zeilen der Abbildung stanzt natürlich das Rechteck die Ellipse aus, da hier die Grafikattribute unverändert auf dem Default-Wert (aussparend) belassen wurden.

Letztendlich müssen transparente Objekte für die Druckaufbereitung immer flachgerechnet werden. Die Frage ist nur, an welcher Stelle im Workflow das geschieht. Es kann bereits frühzeitig bei der PDF-Erstellung oder aber auch erst im *Raster Image Processor* (RIP) kurz vor der Plattenbelichtung geschehen. Flachgerechnete Transparenzen führen aber unter Umständen zu Mängeln beim anschließenden Trapping oder Farbmanagement. Insofern ist man bestrebt, das Flachrechnen im zeitlichen Ablauf möglichst nach hinten zu schieben. Auf der anderen Seite können auch beim Flachrechnen Fehler auftreten. Dies geschieht beispielsweise in RIPs, die intern noch auf PostScript beruhen. Da nach dem RIP-Vorgang die Seiten aber in der Regel nicht mehr geprüft werden, wird man in diesem Fall die Transparenzreduzierung eher früher im Workflow vornehmen, um sie anschließend noch einmal visuell zu kontrollieren.

1.2.4 Koordinaten (transformationen)

PDF ist ein geräteunabhängiges Dateiformat. Alle Koordinaten werden in Benutzerkoordinaten (*user space coordinates*) angegeben. Diese werden standardmäßig in DTP-Punkten (1/72 Zoll) ausgedrückt. Diese Benutzerdaten müssen letztendlich vor der Ausgabe in Gerätekoordinaten (*output device units*) umgerechnet werden. Aber auch die einzelnen grafischen Objekte (wie Bilder, Grafiken und Textblöcke), die auf eine Seite platziert werden, haben ihr eigenes Koordinatensystem. Um die Objekte nun auf eine Seite zu positionieren, sie auch gegebenenfalls zu skalieren oder zu drehen, müssen Koordinatentransformationen durchgeführt werden. Diese werden in PDF (wie in PostScript) durch die sogenannte *Current Transformation Matrix* (CTM) beschrieben. Die Mathematik dieser sehr grundlegenden Transformation (nicht nur für PDF, sondern auch für andere Formate) soll an dieser Stelle kurz erläutert werden.

Man könnte vermuten, dass eine 2 × 2-Matrix ausreicht, um einen Punkt (x,y) zu transformieren, das heißt, die oben beschriebenen Operationen durchzuführen. Tatsächlich löst aber eine solche Multiplikation

$$(x\ y)\begin{pmatrix} a_{11}\ a_{12} \\ a_{21}\ a_{22} \end{pmatrix} = \begin{pmatrix} a_{11}x + a_{21}y \\ a_{12}x + a_{22}y \end{pmatrix}$$

nicht die geforderten Aufgaben. Denn der Nullpunkt wird wieder auf den Nullpunkt transformiert (x = y = 0), so dass keine freie Positionierung der einzelnen Objekte möglich ist.

Der Trick ist, den zweidimensionalen Punkt (x,y) formal als einen Punkt (x,y,1) im dreidimensionalen Raum zu schreiben und dann eine geeignete 3 × 3-Matrix für die Transformation zu definieren:

$$(x\ y\ 1)\begin{pmatrix} a_{11}\ a_{12}\ 0 \\ a_{21}\ a_{22}\ 0 \\ a_{31}\ a_{32}\ 1 \end{pmatrix} = \begin{pmatrix} a_{11}x + a_{21}y + a_{31} \\ a_{12}x + a_{22}y + a_{32} \\ 1 \end{pmatrix}$$

Das Ergebnis rechts kann man dann wieder als einen zweidimensionalen Punkt interpretieren, in dem man die dritte Koordinate, die ja konstant den Wert 1 hat, einfach wieder weglässt. Insgesamt hat man also eine Transformation

$$(x\ y) \longmapsto \begin{pmatrix} a_{11}x + a_{21}y + a_{31} \\ a_{12}x + a_{22}y + a_{32} \end{pmatrix}$$

welche durch die Werte ($a_{11}\ a_{12}\ a_{21}\ a_{22}\ a_{31}\ a_{32}$) vollständig beschrieben wird. Die dritte Spalte der 3 × 3-Matrix enthält nur konstante Elemente und wird deswegen nicht mit aufgeführt. Eine CTM besteht also aus einem Feld von nur 6 Werten.

Man beachte, dass durch die so definierte Transformation der Nullpunkt (0,0) auf einen beliebigen Punkt (a_{31},a_{32}) verschoben werden kann.

1.2.5 Transferfunktionen

Für die Druckausgabe werden Tonwerte im RIP noch einmal verändert. Ziel dieser Tonwertanpassungen kann beispielsweise die Kompensation der aktuellen Tonwertzunahmen[6] im Druck auf vorgeschriebene Standardwerte sein. Die Gründe und auch das praktische Vorgehen werden in Abschn. 4.4.3 genauer diskutiert. Es ist festzuhalten, dass Transferfunktionen geräteabhängig sind. Denn die Ausprägung einer solchen Tonwertkompensationskurve hängt dabei von einer Vielzahl von Faktoren ab, wie Druckmaschine, Bedruckstoff, Farben,

[6] Gedruckte Punkte erscheinen in der Regel größer, als sie in den digitalen Daten definiert sind. Diese Differenz nennt man Tonwertzuwachs oder Tonwertzunahme. Ein negativer Tonwertzuwachs wird auch als Tonwertabnahme bezeichnet.

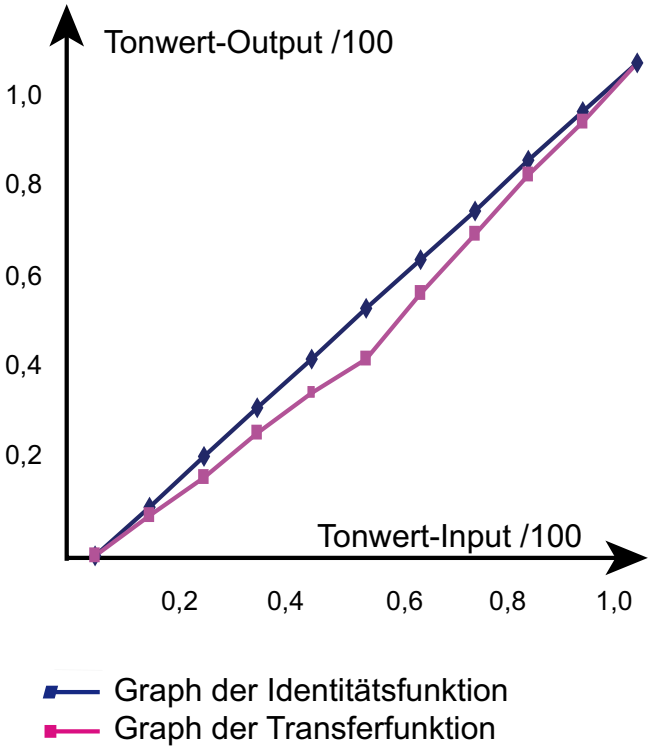


Abb. 1.10 Tonwerte werden durch Transferkurven modifiziert – in diesem Beispiel verringert

CtP-Belichter, Druckplatten, aber auch vom Rastersystem und dem angestrebten Standard. Folglich werden Transferkurven normalerweise erst sehr spät im Workflow in eine PDF-Datei geschrieben (bei der Plattenausgabe in der Druckerei) oder aber auch immer mehr in einem Metadatenformat formuliert und damit außerhalb von PDF dem RIP zugeführt.

Solche Tonwertmodifikationen sind nicht linear, können also nicht durch die einfache Multiplikation mit einem Faktor beschrieben werden. Tatsächlich lässt sich im Allgemeinen keine analytische Funktion finden, welche die Tonwertanpassung definiert. Stattdessen wird – vereinfacht gesagt – eine zweispaltige Tabelle verwendet, die empirisch ermittelt werden muss. Der originale Tonwert entspricht dem ersten Spalteneintrag und der veränderte Tonwert dem zweiten. Um nicht beliebig viele Listeneinträge angeben zu müssen, werden nur gewisse Stützstellen definiert und für alle dazwischen liegenden Eingangswerte die Listenelemente interpoliert. Ist beispielsweise für den Tonwert von 50 % ein Wert von 44 % eingetragen und für den 40 %-Tonwert ein Wert von 36 %, so wird der 45 %-Tonwert auf den Tonwert von (44 + 36) / 2 = 40 % abgebildet (s. Abb. 1.10).

Tonwerte werden üblicherweise von 0 % bis 100 % (= Vollton) gemessen. Eine Tonwertanpassung ist also eigentlich eine Funktion von dem Intervall [0,100] auf das Intervall [0,100]. In PDF wird allerdings anders normiert und die Transferfunktion ist eine Funktion von [0,1] auf [0,1].

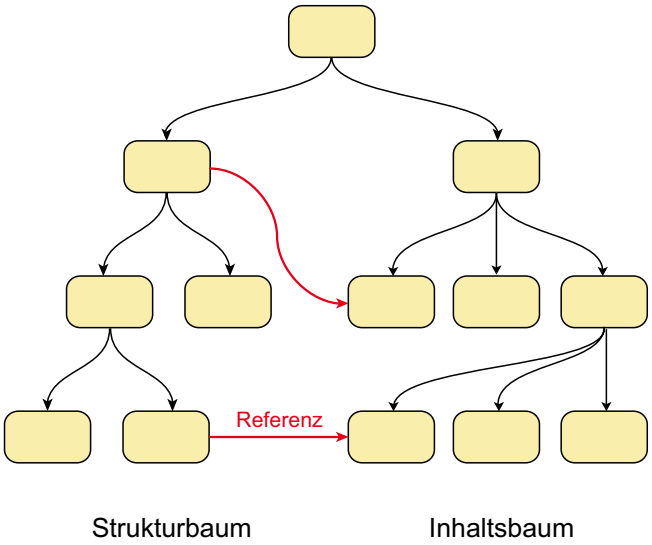


Abb. 1.11 PDF-Dokumente können ab Version 1.4 auch strukturelle Informationen enthalten. Damit können beispielsweise Überschriften oder Fußnoten unabhängig von den Schrifteigenschaften definiert werden

1.2.6 Tagged PDF

PDF war ursprünglich nur ein Dokumentenformat, in dem das Layout von Seiten beschrieben werden konnte. Es genügte also, beispielsweise Dokumentelemente wie Überschriften, den eigentlichen Text, Fußnoten, Seitenzahlen und dergleichen anders zu formatieren, um dann in der Ausgabe für den Leser deutlich zu machen, um was es sich handelt.

Seit der PDF-Version 1.4 können aber PDF-Dateien zusätzlich noch strukturelle Informationen enthalten. Diese logischen Strukturen definieren dann Dinge wie Überschrift, Absatz, Tabellen, Zitate und sind zunächst völlig unabhängig von den sichtbaren Inhalten der PDF-Datei. Natürlich müssen die Strukturelemente einen Bezug auf die Inhalte herstellen. Diese logischen Strukturen werden durch *Tags* definiert, wobei das englische Wort *Tag* so viel wie „Schildchen“ oder „Etikett“ bedeutet. Daher also der Ausdruck *Tagged PDF*.

Die strukturelle Information wird baumartig geordnet. Von einem Wurzelelement *StructTreeRoot* lassen sich die verschiedenen Strukturelemente *StructElem* und Unterstrukturelemente erreichen. Man spricht deswegen auch von einem (semantischen) Strukturbaum. Auch die eigentlichen Inhaltsdaten werden baumartig aufgebaut, wie der nächste Abschnitt zeigt. Beide Teilbäume (genauer: Äste) sind bis auf die Referenzen unabhängig voneinander (s. Abb. 1.11).

Doch für was wird ein solcher Strukturbaum eigentlich benötigt? An dieser Stelle sollen drei Situationen beschrieben werden, bei denen solche Strukturdefinitionen von Vorteil sind:

Barrierefreies PDF: Für Sehbehinderte können Texte im PDF mit geeigneter Software (beispielsweise mit Acrobat) auch vorgelesen werden. Damit das vernünftig funktioniert,

muss die Lesereihenfolge definiert sein (man denke beispielsweise an Fußnoten), Textteile, die nicht vorgelesen werden sollen (Seitenzahlen, Kopf- oder Fußzeilen), müssen speziell markiert werden, und Bilder sollten textuelle Ersatzbeschreibungen erhalten. Eine Voraussetzung ist außerdem die Textcodierung im Unicode-Format (s. Abschn. 2.1.1). Ansonsten kann man beispielsweise nämlich nicht zwischen einem Minuszeichen, einem Bindestrich und einem Trennungsstrich differenzieren. Barrierefreie Dokumente werden in Abschn. 3.5 behandelt.

Dynamische Umformatierungen (*re-flow*): Für Lesegeräte mit sehr limitierten Bildschirmgrößen (PDA, E-Book, Mobiltelefon, …) sollte Text in einer PDF-Datei neu umbrochen werden. Ansonsten muss bei jeder breiteren PDF-Seite ständig mit der Bildlaufleiste gescrollt werden. Um aber eine dynamische Umformatierung zu ermöglichen, muss nicht nur die Lesereihenfolge klar geregelt sein, sondern auch, welcher Text beispielsweise zu einem Absatz gehört. Man beachte, dass Text in PDF sogar buchstabenweise einzeln platziert werden kann. In jedem Fall kann man nicht davon ausgehen, dass ein Absatz oder auch nur eine Zeile oder ein Wort in einer PDF-Datei zusammen in einer Unterstruktur enthalten sind.

Konvertierung in andere Dateiformate: Soll ein PDF in ein anderes Dateiformat wie HTML konvertiert werden, so werden Strukturinformationen benötigt. Wenn also PDF ein Kandidat für ein Dateiformat im Bereich Cross Media Publishing sein möchte, muss es diese logischen Strukturinformationen geben.

Doch wie kommen die Strukturinformationen in eine PDF-Datei hinein? Es gibt mehrere Möglichkeiten, die an dieser Stelle aber nur kurz angesprochen werden sollen.

Zunächst kann die Strukturierung quasi manuell auf Grundlage einer PDF-Datei erfolgen. So lassen sich im Programm *Acrobat* neue Tags definieren, aber auch vorhandene Tags editieren und Tags den Inhalten zuordnen. Dieses Verfahren ist wohl weniger geeignet, ein komplexes Dokument vollständig zu strukturieren, als vielmehr ein bereits *getagtes* PDF zu überprüfen und gegebenenfalls zu korrigieren.

Tagged PDF entsteht vielmehr eher durch die Konvertierung bereits vorab strukturierter Daten. Die Struktur kann beispielsweise indirekt durch Formatvorlagen in Layoutprogrammen festgelegt werden oder aber auch direkt durch HTML- oder XML-Editoren. Die festgelegten logischen Strukturen werden dann bei der PDF-Konvertierung in *PDF-Tags* übernommen.

1.2.7 Externe Objekte

In einer PDF-Datei können auch autonome Datenströme mit grafischem Inhalt definiert werden. Diese Objekte werden als „extern“ bzw. als *XObjects* bezeichnet, obwohl sie auch innerhalb der Datei liegen können, welche die *XObjects* verwendet. Letzteres geschieht mit dem Operator *Do*. Seit der PDF-Version 1.4 (s. Abschn. 1.2.1) können aber die *XObjects* auch von externen Dateien referenziert werden, so dass eine Auslagerung von grafischen Inhalten möglich ist.

Es gibt drei verschiedene Arten von *XObjects*:

- Ein *Image XObject* beschreibt ein digitalisiertes Halbtonbild.
- Ein *PostScript XObject* enthält Fragmente von PostScript-Code.
- Ein *Form XObject* kann eine Folge beliebiger grafischer PDF-Objekte sein (Texte, Bilder, Grafiken). Die ursprüngliche Idee war, damit Formulare nur ein einziges Mal in einer PDF-Datei zu beschreiben, um anschließend auf mehreren Seiten nur noch Einträge der Formularfelder an den richtigen Positionen zu speichern. Man bemerke, dass diese Formulare nichts mit den in der PDF-Anwendung beliebten interaktiven Formularen zu tun haben.

1.2.8 PDF-Dateistruktur

Während PostScript als Programmiersprache eigentlich dafür ausgelegt ist, Zeile für Zeile interpretiert zu werden, ist PDF als Dokumentenformat gänzlich anders strukturiert. Die Einzelteile eines PDF-Dokumentes werden nämlich in „Objekte“ verpackt, auf die ein beliebiger Zugriff möglich ist. Letzteres wird gewährleistet, da jede PDF-Datei eine Art Inhaltsübersicht (*Cross-Reference Table*) enthält, die besagt, wo sich die einzelnen Objekte in der Datei befinden. Damit fällt es einer PDF-verarbeitenden Software leicht, alle benötigten Teile einer oder mehreren Seiten einzusammeln. Bei PDF können also Seiten unabhängig voneinander verarbeitet werden, was bei PostScript nicht immer möglich ist.

Die Abb. 1.12 zeigt die generelle baumartige Struktur eines PDF-Dokumentes, wobei nur wenige Objekte eingezeichnet sind. Die Wurzel stellt das Objekt *Catalog* dar (der Baum wächst also in botanisch unüblicher Weise von links nach rechts). Dieses Objekt ist der Ausgangspunkt von vielen unterschiedlichen Ästen. Hier sind nur vier aufgeführt: *Pages, Outlines, Metadata* und *StructTreeRoot.*

Das *Pages*-Objekt enthält Referenzen auf die einzelnen Seiten des Dokuments, das heißt also auf alle Objekte des Typs *Page*. In diesen steckt nun die Beschreibung einer Seite. Beispielsweise sind dort (optional) Beschnitt- und Endformat aufgeführt. In Abb. 1.12 sind nur Referenzen auf drei weitere Objekte eingezeichnet: In *Contents* wird der sichtbare bzw. druckbare Inhalt der Seite aufgeführt. Ohne ein *Contents*-Objekt ist eine Seite also leer. Im *Thumb* ist ein Voransichtsbild (*thumbnail*) der Seite gespeichert und in *Annots* befinden alle Kommentare (*annotations*), die zu dieser Seite gemacht wurden.

In *Outlines* hingegen sind die Lesezeichen zusammengefasst, jedes *Outline Item* beschreibt genau ein Lesezeichen.

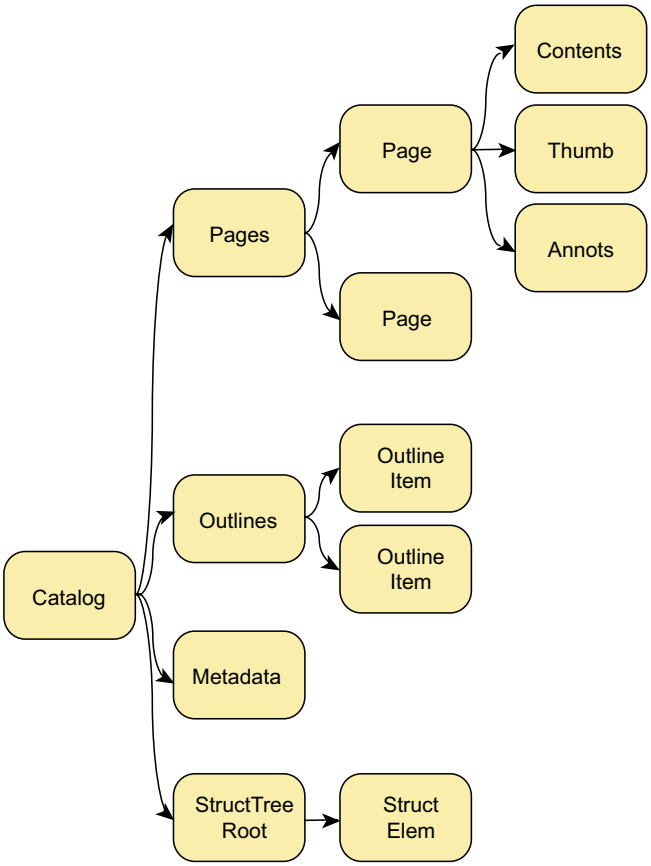


Abb. 1.12 PDF-Dokumnete werden durch eine baumartige Struktur von Objekten beschrieben

In einem *Metadata*-Objekt werden Zusatzangaben über das PDF-Dokument gemacht, die über die eigentlichen Druckdaten hinausgehen wie Autorenname, Urheberrechte usw. (s. Abschn. 2.6). In *StructTreeRoot* befindet sich der semantische Strukturbaum, der in Abschn. 1.2.6 vorgestellt wurde.

Die Abb. 1.13 zeigt die typische Struktur eines Objekts, das von den beiden Schlüsselwörtern *obj* und *endobj* eingerahmt wird. Die erste Ziffer vor dem *obj* ist die Objekt-ID, die zweite die sogenannte Generationsnummer (*generation number*). Jedes Objekt erhält ein innerhalb des PDF-Dokumentes eindeutiges Zahlenpaar von Objekt-ID und Generationsnummer, mit Hilfe derer das Objekt leicht von anderen Objekten referenziert werden kann. Auch das Auffinden des Objekts mittels der oben genannten *Cross-Reference Table* funktioniert über diese beiden Zahlen.

Die beiden spitzen Klammern „<<“ und „>>“ bezeichnen Anfang und Ende eines Wörterbuchs (*dictionary*). Ein solches *Dictionary* ist eine Tabelle von Objektpaaren wie */Type* und */Pages*. Dieser *Dictionary*-Eintrag besagt, dass das Objekt von Typ *Pages* ist. Das Beispiel zeigt also den „Ausgangspunkt“ aller Seiten. Wie viele Seiten mit welcher Objekt-ID sich in diesem PDF-Dokument befinden, zeigt der nächste *Dictionary*-Eintrag. Man sieht, dass dort die *Kids* aufgelistet sind, womit die Seiten gemeint sind. Das *R* in der eckigen Klammer steht für *Referenz*, also für eine Beziehung zu einem anderen Objekt. In dem Beispiel ist *6 0 R* nämlich eine Referenz auf das Objekt mit der ID gleich 6. In dem Beispiel hat folglich das PDF-Dokument drei Seiten (wegen der drei Referenzen). Die eckigen Klammern bedeuten eine Reihe (ein Feld) von Einträgen. Hier sind es drei Tripel, nämlich 6 0 R, 8 0 R und 12 0 R.

Meist am Ende einer PDF-Datei befindet sich die *Cross-Reference Table*, die das Auffinden der Objekte innerhalb der Datei erleichtert. Abbildung 1.14 gibt ein Beispiel einer solchen Tabelle wieder. Das Schlüsselwort *xref* markiert den Anfang der Tabelle. Jede *Cross-Reference Table* zeigt die Positionen von Objekten, die einen zusammenhängenden Bereich von Objekt-IDs haben. In dem vorliegenden Beispiel erkennt man in der zweiten Zeile, dass es sich um Objekte handelt, deren IDs von 0 an zählen. Insgesamt sind es 6 unterschiedliche Objekte, das heißt folglich, die IDs zählen von 0 bis 5.

Die folgenden sechs Zeilen geben die Position der einzelnen Objekte innerhalb der Datei an – die dritte Zeile also die Position des ersten Objekts mit der ID gleich 0, die vierte Zeile die Position des zweiten Objekts mit der ID gleich 1 und so weiter. Die Position wird in Byte-Abstand von Anfang der Datei angegeben und befindet sich in jeder Zeile als erster Eintrag (von links). Der letzte Eintrag in der Zeile besagt, ob das Objekt im Gebrauch (*n*) oder bereits gelöscht (*f*) ist. Da jede *Cross-Reference Table* einen zusammenhängenden Bereich von IDs verwaltet, müssen auch die freien Einträge noch aufgelistet werden. Die fünfziffrigen Zahlen in den unteren Zeilen sind die sogenannten *Generation Numbers*. Sie sind recht technischer Natur und werden hier nicht weiter ausgeführt.

Das PDF-Format wurde speziell für den Druck variabler Daten (personalisiertes Drucken) erweitert und ist unter dem Namen PDF/VT bekannt (ISO 2010). Bei PDF/VT geht es vor allem darum, dass wiederkehrende und personalisierte Inhaltskomponenten von Seiten unterschiedlich verarbeitet werden sollten. Das Format wird in Abschn. 4.4.5 genauer untersucht.

1.2.9 Alternative Dokumentenformate

2005 hat die Firma Microsoft das damals neue Dateiformat *XML Paper Specification* (*XPS*) vorgestellt. Programme zum Anzeigen (*XPS-Viewer*) und zum Drucken (*XPS-Writer*) werden seit *Windows Vista* mit ausgeliefert, können aber auch für ältere Windows-Versionen nachinstalliert werden. Microsoft übergab die Spezifikation der *ECMA International* (ECMA steht für *European Computer Manufacturers Association*), woraus sich das *Open XPS* entwickelte und 2009 als ECMA-Standard 388 (ECMA 2009) verabschiedet wurde.

XPS ist dem PDF-Format in einigen Eigenschaften ähnlich. Insbesondere können Seitenlayouts geräteunabhängig

Abb. 1.13 Beispiel für den Aufbau eines PDF-Objekts

```
xref
0 6
0000000003  65535 f
0000000017  00000 n
0000000081  00000 n
0000000000  00007 f
0000000331  00000 n
0000000409  00000 n
```

Abb. 1.14 Die Cross-Reference Table enthält Positionsangaben der PDF-Objekte. Das Beispiel wurde aus der PDF Spezifikation 1.7, Seite 96, entnommen

beschrieben werden. Auch werden Grafiken in XPS wie in PDF durch Bézierkurven definiert, die mit Verläufen, Mustern oder Farben gefüllt werden, OpenType-Schriften können in XPS-Dokumente integriert werden und auch Hyperlinks werden unterstützt.

Allerdings bietet XPS derzeit deutlich weniger Multimedia-Unterstützung als PDF. Auch wird für den Druckbereich die Eigenschaft der Transparenz unterstützt, nicht jedoch das „Überdrucken“. Insofern hat XPS bisher noch keine große Marktdurchdringung im professionellen Publishing-Bereich erworben.

Jedes XPS-Dokument stellt ein ZIP-Archiv dar. Das wiederum ist ein Container-Format, in dem verschiedene Verzeichnisse und Dateien in einer Struktur zusammengefasst und komprimiert werden. Mit Hilfe eines Programms, das ZIP-Archive wieder entpackt, kann man leicht die genaue Struktur eines XPS-Dokumentes analysieren.

Ein XPS enthält einen *Document*-Ordner, der seinerseits wieder Unterordner besitzt. In dem Unterordner *Pages*[7] befinden sich die einzelnen Seiten, nämlich einerseits die Beschreibung der Inhalte (*FPage*-Dateien, die ihrerseits *FixedPage*-Elemente enthalten) und auch die Beziehungen (*_rels*) dieser Seiten zu benötigten Ressourcen wie Schriften und dergleichen. Einen beispielhaften Aufbau eines XPS-Dokuments, das drei Seiten, zwei Schriften und vier Bilder enthält,

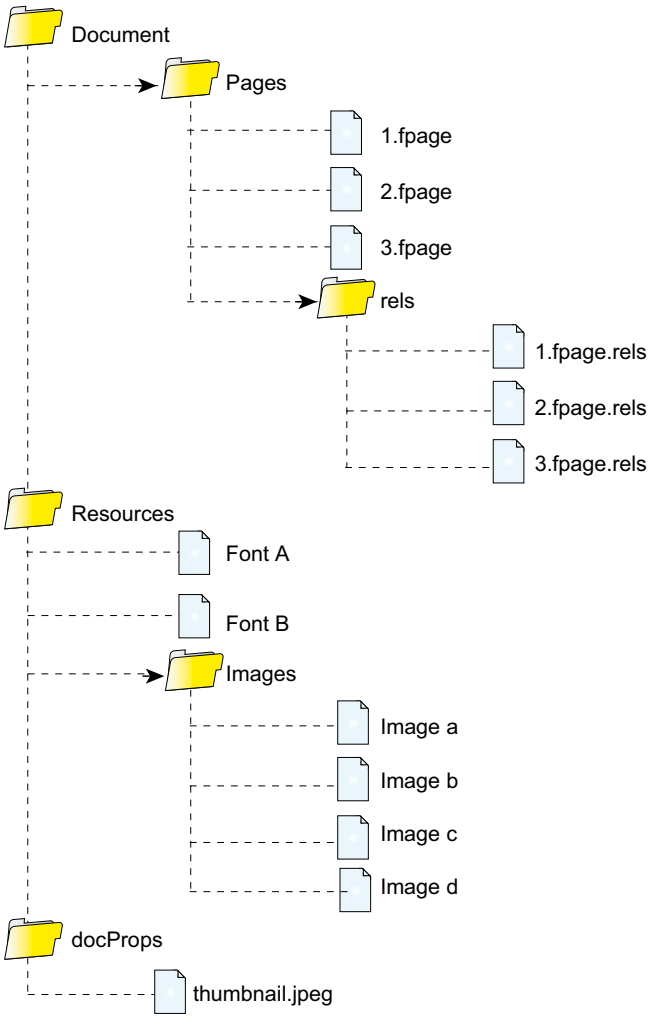


Abb. 1.15 Aufbau einer XPS-Datei

ist stark vereinfacht in Abb. 1.15 dargestellt. Die Pfeile zeigen jeweils auf einen Unterordner.

Eine *FPage*-Datei enthält dann neben Angaben über beispielsweise die Seitengröße vor allem auch die Inhalte, die angezeigt oder ausgedruckt werden sollen, wie beispielsweise die Zeichencodes. Die *FPage*-Datei ist – wie die meisten Dateien eines XPS-Dokuments – in XML codiert.

Auf der Stufe des *Document*-Ordners befindet sich auch der *Resources*-Ordner, in dem Fonts und Bilder gespeichert sind. Die Bilder selber können dann im JPEG-, TIFF- oder PNG-Format codiert sein, als eingebettete Schrift ist nur das *OpenType*-Format zulässig.

[7] Eigentlich enthält der *Document*-Ordner zunächst einen oder mehrere Dokumentenordner, von denen jeder einen Unterordner *Pages* besitzt.

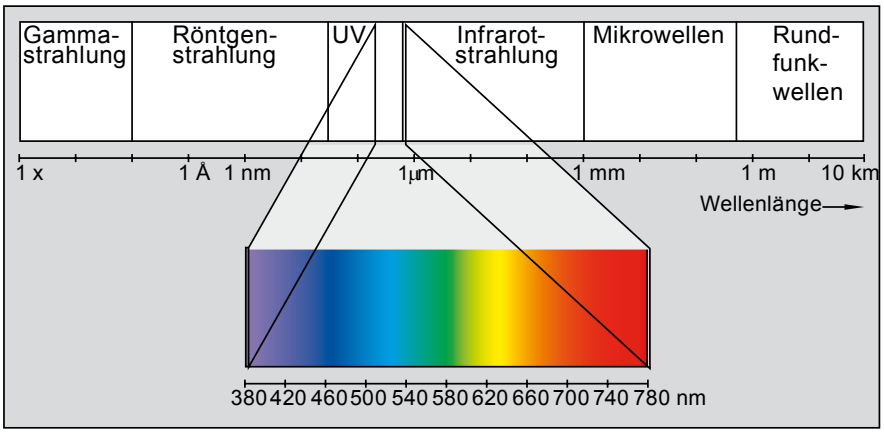


Abb. 1.16 Spektrum elektromagnetischer Strahlen

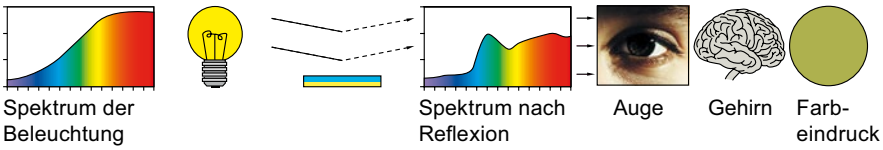


Abb. 1.17 Farbwahrnehmung von der Beleuchtung bis zum Gehirn

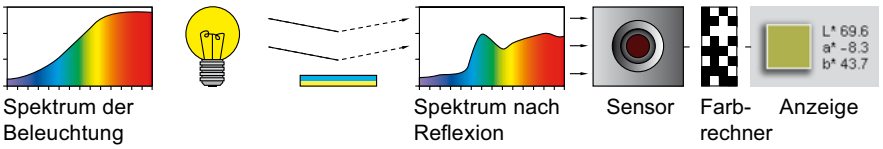


Abb. 1.18 Spektralphotometer von der Beleuchtung bis zum Farbwert

Der Ordner *docProps* enthält ein Vorschaubild für die erste Seite. Darüber hinaus gibt es im Allgemeinen aber noch weitere Unterordner und Dateien in einem XPS-Dokument, die hier aber nicht weiter ausgeführt werden.

1.3 Einführung Farbe

Licht ist elektromagnetische Strahlung verschiedener Wellenlängen, von denen das Auge einen Bereich von ca. 380 bis 780 nm wahrnehmen kann. Lichtstrahlen einer einzigen Wellenlänge werden als reines, farbiges Licht wahrgenommen. Abbildung 1.16 zeigt die Zuordnung von Wellenlängen und der wahrgenommenen Farbe. Übliche Lichtquellen enthalten Strahlung verschiedener Wellenlängen, die in ihrer Gesamtheit Spektrum genannt werden. Überwiegen die blauen Anteile im Spektrum des Lichts, so wird dieses als kaltes Licht wahrgenommen und auch beleuchtete Gegenstände erscheinen in kühleren Farben. Überwiegt der gelb-rote Anteil im Spektrum, so spricht man von einem warmen Licht.

Ein vereinfachtes Modell der Farbwahrnehmung kann als Verarbeitungskette dargestellt werden, die mit dem Spektrum der Beleuchtung beginnt. Dies fällt auf einen Gegenstand, der Teile des einfallenden Lichts absorbiert. Das reflektierte Spektrum aktiviert im Auge entsprechende Rezeptoren, deren Signale durch die anschließende Verarbeitung im Gehirn eine Farbwahrnehmung hervorrufen (s. Abb. 1.17).

Ein Spektralphotometer arbeitet ähnlich und besitzt eine Lichtquelle mit einem bekannten Spektrum. Bei der Messung absorbiert die Probe Teile des Spektrums der Beleuchtung. Das reflektierte Spektrum wird dann von einem Sensor aufgenommen, der die Intensitäten der verschiedenen Wellenlängen in Zahlenwerte umwandelt. Das Ergebnis dieses Messvorganges sind spektrale Messwerte. Die spektralen Messwerte werden in Farbwerte umgerechnet (s. Abb. 1.18).

Typische Spektralphotometer können oft sowohl Farben auf Drucken als auch auf Monitoren messen.

1.3.1 Die Farbmodelle Grau, RGB, CMYK und CIE L*a*b*

Farben lassen sich mit verschiedenen Modellen beschreiben. Welches Modell zum Einsatz kommt, hängt sowohl vom Einsatzzweck als auch von den Möglichkeiten der genutzten Programme ab. Dabei wird zwischen geräteabhängigen Farbmodellen und geräteunabhängigen Farbmodellen unterschieden. Das geräteabhängige Farbmodell beschreibt Farben, die durch die direkte Ansteuerung eines Gerätes entstehen. Geräteabhängige RGB-Farben für einen LCD-Monitor ergeben sich beispielsweise durch die elektrischen Felder, welche die Flüssigkeitskristallmoleküle in ihren Ausrichtungen ändern.

Abb. 1.19 Schematische Darstellung der Farbmischung im RGB-Modell

Abb. 1.20 Schematische Darstellung der Farbmischung im CMY-Modell

Bei Monitoren unterschiedlicher Bauweise können gleiche Gerätefarbwerte visuell unterschiedliche Farbeindrücke ergeben. Gleiches gilt zum Beispiel für die Ansteuerung eines Drucksystems mit CMYK-Farben. Je nach Drucksystem und verwendetem Papiertyp führen die gleichen Gerätefarbwerte zu unterschiedlichen visuellen Eindrücken.

Ein geräteunabhängiges Farbmodell beschreibt Farben unabhängig von einem bestimmten Gerät und kommt auch bei der Farbmessung zum Einsatz.

Das additive RGB-Modell

Am meisten verbreitet ist das RGB-Modell, das auf den Grundfarben Rot, Grün und Blau beruht. Es repräsentiert die sogenannte additive Farbmischung für selbst leuchtende Systeme wie zum Beispiel Monitore. Jeder Farbton im RGB-Farbsystem wird dabei durch den Anteil der Grundfarbe dargestellt. Üblich ist dabei eine Skala von 0–255 Abstufungen (8 Bit) pro Farbton.

Mischt man zwei Grundfarben, so addieren sich diese zu einem helleren Farbton. Leuchten alle Grundfarben gleichzeitig mit voller Stärke addieren sie sich zu Weiß. Leuchtet keine der Grundfarben, so ergibt dies ein Schwarz.

Neutrale oder graue Farbtöne zeichnen sich durch gleiche Anteile der drei RGB-Grundfarben aus.

Die Mischtöne von jeweils zwei RGB-Grundfarben sind Cyan, Magenta und Gelb. Abbildung 1.19 zeigt eine schematische Darstellung der Farbmischung im RGB-Modell.

Das subtraktive CMY (K)-Modell

Das subtraktive CMY-Modell beschreibt die Farbmischung bei der fotografischen Entwicklung oder der Druckausgabe. Es funktioniert genau umgekehrt zum RGB-Modell. Voraussetzung für die Farbmischung sind eine Lichtquelle und ein weißes Trägermedium (z. B. Papier), auf dem die subtraktiven Farben ausgemischt werden. Ist die Intensität aller Grundfarben gleich null, so wird das Weiß des Druckmediums vollständig wiedergegeben.

Sind die drei Druckfarben mit voller Intensität vorhanden, so absorbiert jede der Grundfarben ein Teil des einfallenden Lichtes und man sieht eine schwarze Fläche.

Je stärker eine Grundfarbe aufgetragen wird, desto mehr Farbe vom einfallenden Licht wird absorbiert. Vom Weiß wird sozusagen Farbe subtrahiert. Die Mischtöne von jeweils zwei CMY-Grundfarben sind Rot, Grün und Blau. Neutrale oder graue Farbtöne ergeben sich in einem idealisierten Modell aus gleichen Anteilen aller drei CMY-Grundfarben.

Die vierte Farbe Schwarz kommt speziell beim Drucken zum Einsatz, da sich die realen CMY-Grundfarben aufgrund der verfügbaren Farbpigmente nicht so verhalten wie im idealen Modell. So ergibt die Mischung aller Farben meist kein Schwarz, sondern oft nur ein Dunkelbraun. Die Hinzunahme von Schwarz in der Farbmischung ermöglicht die Darstellung von Tiefschwarz. Weiterhin lassen sich neutrale Farbtöne mit Schwarz sicherer drucken, und schwarzer Text, der nur mit einer statt mit drei Farben gedruckt wird, ist schärfer und besser lesbar. Abbildung 1.20 zeigt eine schematische Darstellung der Farbmischung im CMY-Modell.

Grau

Das Farbmodell für Grautöne kommt zum Einsatz, wenn ein Ausgabegerät nur Grautöne und keine bunten Farben darstellen kann.

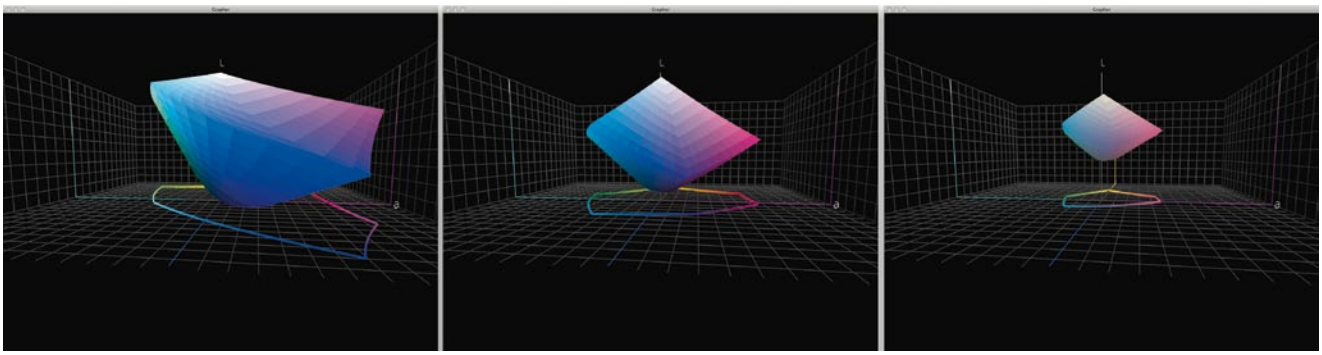


Abb. 1.21 Unterschiedlicher Farbräume (Monitor, Fotodruck, Zeitungsdruck) im Lab-Modell

CIE L*a*b*-Modell

Betrachtet man die reinen Grundfarben auf verschiedenen Monitoren oder Drucksystemen, so zeigen diese durchaus visuell sichtbare Unterschiede. Das Gleiche gilt auch für den Vergleich von Mischfarben mit gleichen Anteilen.

Um solche Unterschiede beschreibbar und messbar zu machen, wurde von der **International Commission on Illumination CIE**[8] – einer Vereinigung von Farbwissenschaftlern – das geräteunabhängige CIE L*a*b*-Farbmodell entwickelt, welches auf der systematischen Auswertung von Experimenten zum menschlichen Farbsehen beruht.

Jede Farbe lässt sich im CIE L*a*b*-Modell durch die Komponenten L* (Helligkeit), a* (Intensität auf der Grün-Rot-Achse) und b* (Intensität auf der Blau-Gelb-Achse) beschreiben.

Gleiche geometrische Abstände von zwei Farbwerten im CIE L*a*b*-Modell entsprechen weitgehend auch visuell wahrgenommenen gleichen Farbabständen. Daher spricht man auch von einem gleichabständigen Farbraum. Eine Vertiefung zum CIE L*a*b*-Modell findet sich in Abschn. 2.5. In fast allen Anwendungsprogrammen der grafischen Industrie wird die wissenschaftlich korrekte Bezeichnung CIE L*a*b* auf Lab eingekürzt, was teilweise aus Platzgründen in Menüs und Grafiken hilfreich ist. Dies ist auch in Grafiken, die in diesem Buch verwendet werden, der Fall.

Vom Farbmodell zum Farbraum

Das CIE L*a*b*-Modell spannt ein Koordinatensystem auf, in dem alle Farben eines Ausgabesystems (z. B. Monitor oder Drucker) dargestellt werden können. Dabei ergibt sich ein räumliches Gebilde, das als Farbraum bezeichnet wird (s. Abb. 1.21). So gibt es zum Beispiel Farbräume verschiedener Monitore oder Drucker. Monitore, die sehr intensive Farben darstellen können, haben einen größeren Farbraum als jene, deren Wiedergabe blässlicher ist. Das Gleiche gilt für den Druck auf unterschiedlichen Papieren. Auf Zeitungspapier kann nur ein kleinerer Druckfarbraum realisiert werden als auf einem hochwertigen Fotopapier.

[8] www.cie.co.at, Stand 24.02.2013.

1.3.2 Farbprofile für Geräte und Austauschfarbräume

Mittels Farbprofilen lassen sich Unterschiede in der Farbumsetzung verschiedener Ein- und Ausgabegeräte minimieren. Sehr stark vereinfacht ist ein Farbprofil eine große Tabelle, die alle geräteabhängigen Farbwerte eines Ein- oder Ausgabegerätes mit geräteunabhängigen CIE L*a*b*-Farbwerten verknüpft.

Stehen in einem Farbsystem für die RGB-Grundfarben 256 Abstufungen (8 Bit) zur Verfügung, so hätte eine Tabelle für sämtliche Mischfarben 256^3 also ca. 16,7 Millionen Einträge. In einem Farbprofil sind daher nur eine Auswahl von RGB-Farben und ihren entsprechenden CIE L*a*b*-Farbwerten enthalten, wie man in Abb. 1.22 sieht.

Statt eines realen Ein- oder Ausgabegerätes kann ein Farbprofil auch einen sogenannten Austauschfarbraum repräsentieren. Austauschfarbräume spielen im Farbmanagement eine wichtige Rolle, da sie Farbräume für den Austausch von digitalen Bildern bzw. kompletten Designdaten festlegen. Speziell im Designprozess, wenn Farben numerisch angelegt werden, kommt dafür ein Austauschfarbraum zum Einsatz. Liegen auch die Bilddaten in einem branchenüblichen Austauschfarbraum vor, so lassen sich Text, Grafik und Bild einfach zu einem Dokument in einem einheitlichen Farbraum kombinieren.

Die wichtigsten Austauschfarbräume sind der sRGB-Farbraum, der einen standardisierten Monitor beschreibt, und der CMYK-Farbraum für den Offsetdruck auf gestrichenem (engl.: *coated*) Papier. Letzterer ist im Standard ISO 12647-2 definiert (s. Abschn. 4.1). Daher wird für diesen Austauschfarbraum auch das Kürzel *ISOcoated_v2* verwendet. Abbildung 1.23 zeigt Profilsymbole für den Austauschfarbraum sRGB, für einen Monitor, für einen Drucker und den Austauschfarbraum ISOcoated_v2.

Arbeiten im sRGB-Farbraum unter Anwendung von Farbprofilen

Abbildung 1.24 zeigt die prinzipielle Anwendung von Farbprofilen beim Arbeiten im Austauschfarbraum sRGB. Die