Erzincan Üniversitesi
Fen Bilimleri Enstitüsü Dergisi
2021, 14(3), 887-897
ISSN: 1307-9085, e-ISSN: 2149-4584
Araştırma Makalesi

Erzincan University
Journal of Science and Technology
2021, 14(3), 887-897
DOI: 10.18185/erzifbed.792292
Research Article

Development and Implementation of Electronic Library for Protection of Copyrights

Celal ATALAR[1*]

[1]Karadeniz Technical University, Of Technology Faculty, Department of Software Engineering, Trabzon

Geliş / Received: 08/09/2020, Kabul / Accepted: 03/02/2021

Abstract

Today, many different publications such as journals and books are presented in electronic environment. Copyright concept is still discussed about the distribution of the books and paid publications digitally. The main objective of electronic library application to be developed is to display a certain part of the books and publications including the information that the users intend to reach. In this study, such operations as the search of phrases within various portable document format (PDF) documents like e-books, theses and technical reports or on the bookmarks of the documents, and the generation of its results as a PDF document have been showed. It has been viewed that the phrase search process gives more accurate results than the search on bookmarks, but is slower. As a result of the studies conducted on PDF documents specially created according to parameters such as file size, number of pages, content type and number of bookmarks, it has been observed that these parameters affect search performance at different degrees. In addition, an open source plug-in that can perform these operations has been also developed for the electronic library application. The plug-in developed independently of electronic library application in Java programming language can easily be added to the web applications. Apart from the plug-in, there is also a lot of information required for users to create and manage their own PDF documents.

Keywords: Digital Libraries, Information-system Modelling, Support Tools and Languages for Information-System Development, Electronic Libraries, Database and Information-system Integration

Telif Haklarının Korunmasına Yönelik Elektronik Kütüphane Geliştirilmesi ve Uygulanması

Öz

Günümüzde dergi ve kitap gibi birçok farklı yayın elektronik ortamda sunulmaktadır. Kitapların ve ücretli yayınların dijital olarak dağıtımı hakkında telif hakkı kavramı halen tartışılmaktadır. Geliştirilecek elektronik kütüphane uygulamasının temel amacı, kullanıcıların ulaşmak istedikleri bilgiyi içeren kitap ve yayınların belirli bir bölümünü görüntülemektir. Bu çalışmada, e-kitap, tez ve teknik raporlar gibi çeşitli taşınabilir doküman formatı (PDF) belgelerinde veya belgelerin yer imleri üzerinde sözcük grubu aranması ve sonuçlarının PDF belgesi olarak üretilmesi gibi işlemler gösterilmiştir. Sözcük grubu arama işleminin yer imleri üzerinde yapılan aramadan daha isabetli sonuçlar verdiği ancak daha yavaş olduğu görülmüştür. Dosya boyutu, sayfa sayısı, içerik türü ve yer imi sayısı gibi parametrelere göre özel olarak oluşturulan PDF belgeleri üzerinde yapılan çalışmalar sonucunda bu parametrelerin arama performansını farklı derecelerde etkilediği gözlemlenmiştir. Ayrıca, elektronik kütüphane uygulaması için bu işlemleri gerçekleştirebilecek açık kaynaklı bir eklenti de geliştirilmiştir. Java programlama dilinde elektronik kütüphane uygulamasından bağımsız olarak geliştirilen eklenti web uygulamalarına kolaylıkla eklenebilir. Eklentinin yanı sıra, kullanıcıların kendi PDF belgelerini oluşturması ve yönetmesi için gereken birçok bilgi de bulunmaktadır.

Anahtar Kelimeler: Dijital Kütüphaneler, Bilgi-sistem Modellemesi, Bilgi Sistemi Geliştirmek İçin Destek Araçlar ve Diller, Elektronik Kütüphaneler, Veri Tabanı ve Bilgi Sistemi Entegrasyonu

*Corresponding Author: celalatalar@ktu.edu.tr

1. Introduction

Digital libraries, referred to as electronic libraries, have started to be developed in the early 1990s. With the improvements in computer technology, web technology has gained importance and the number of digital libraries has gradually increased. The books and paid publications which are allowed for access have brought about the requirement for protecting copyrights (McCray and Gallagher, 2001).

Although there are many digital libraries that have been developed, many of them only host copyright free publications.

Google Books, launched in 2004, is considered one of the best digital libraries in terms of usability, appearance, speed and huge number of books as well as hosting copyrighted publications (Kumar and Pamula 2020). The system, which initially operated based on the optical character recognition technique, quickly grown with PDF format books provided by the project partners (Na, 2007). The project, which was improved over time, has become able to convert documents such as PDFs that are collected through web crawler at certain times and has added it to the its index as plain text (Illyes, 2021).

Open Library, launched as a project of Internet Archive in 2006, is another important digital library hosting copyrighted publications although their number is low. The archive created from scanned books causes large file sizes as a result of this the slowness in the document view process is felt by the user. Extracting text information from scanned books is achieved by optical character recognition (Hasbrouck, 2020). The project is open source code and has audiobook reading feature in text information.

HathiTrust which launched in 2008 and include digitized documents provided by Google Books, Internet Archive and several local libraries is a large digital library that hosted copyrighted publications as well. Full text extraction and search procedures were built by developers at University of Michigan (Christenson, 2011).

The specified digital libraries do not perform the search process on document in real-time and show to the user several pages or some paragraphs of the copyrighted documents.

In this study, an open source real-time electronic library plug-in has been developed for the users in order to make searches in e-books which are located on a server and to view search results on a web browser. The electronic library application works on documents in portable document format (PDF), which allows digital viewing of documents developed by Adobe (Warnock, 2020). The plug-in which can perform such processes as doing search in a PDF, creating a PDF, adding contents to PDF, displaying produced PDF document on web browser and saving it has been developed with the help of various libraries. This plug-in helps to fulfil three main functions:

- Making searches in PDF
- Presenting the query results acquired from a database table as PDF
- Showing a certain part of PDF

Doing two different searches could be performed inside a PDF. One of these searches is carried out in the content of PDF. The other one is done on bookmarks in PDF. Search in bookmarks is considered faster than other one; however, it provides more limited results.

2. The Application Framework

In this section, we will first examine our application framework that includes electronic library plug-in (Atalar, 2020). In this framework, various technologies have been used besides java.

After the framework review is completed, we will examine our electronic library plug-in in a very detailed way.

How the system performing search within books in electronic libraries function is given in Figure 1. Firstly, the user searches the books in a category by using book search interface. Book search process is performed based on the name of the books registered in database. The list of the books found after the book search process is displayed on web browser so that the user can select the books in which she/he desires to make a search. After the user select the books on the web page and writes the phrase she/he wanted to search, the search process starts.

Electronic books are located on a directory on the server of the system which the user cannot reach through the internet. The reason of this is to prevent the user from accessing to all the books through an internet connection.

After the user has started phrase search process, the system will find the locations of the books on hard disk according to the registers in database, and then phrase search process will be performed inside the electronic books. If the searched phrase is found after search process that the user has made, the books containing it will be listed on web page. The user can view where the phrase locates in the book after she/he chooses any one from the listed books. In order to prevent unauthorized copying of the entire book, the users are allowed to see a certain part of the books where the phrase locates.

Several libraries have been utilized to fulfil particular functions. The framework including structures and functions has been developed in Java programming language. The web interface which provides interaction with the user has been developed in a web technology called javaserver pages (JSP). JSP codes send signals to framework functions.

MySQL database management system has been utilized to save data about the books. System has a database named "pdf" and single table with four columns referred to as "pdf". First column named "id" contains the number of the book and has properties such as unique, auto-incrementing and positive integer. The second column with the name "category" contains the category of book and has a plain text structure. Third column, "path", holds the location of the book on the server. Fourth column called "name" contains the name of PDF document and has a plain text structure as well.

PDFBox library has been used to get the PDF content in the binary file structure as plain text (Apache Software Foundation, 2019). In addition, iText library has been used for some intermediate processes (Lowagie, 2007).

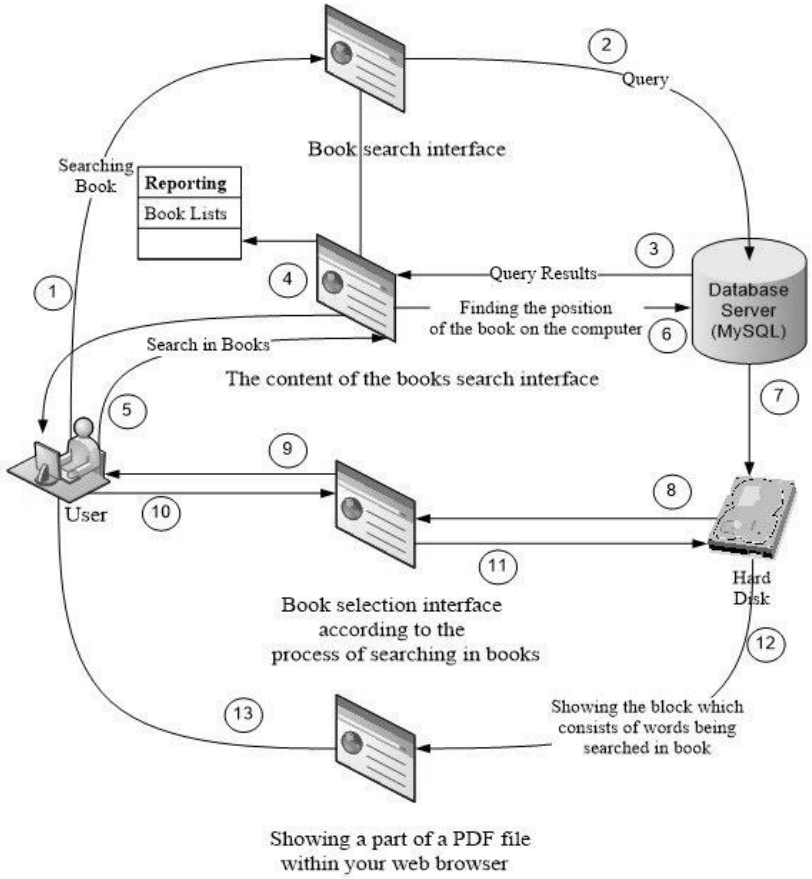

Figure 1. Block diagram of a PDF file search in the system.

2.1. The electronic library plug-in

We have stated above that several functions are developed such as search, reporting and displaying on PDF document. These functions have been gathered under an electronic library plug-in called "PDFMaster.jar" by means of a study which has been developed in Java programming language (Atalar, 2021). Prototypes of the functions within the plug-in are stated below.

- public static int Arama (String PdfPath, String Keyword)
- public static int HizliArama (String PdfPath, String Keyword)
- public static int Raporlama (ResultSet rs)

Arama function given with its prototype is used for searching a phrase in PDF document. First parameter refers to where PDF document is in the computer while second parameter shows the phrase which will be searched in the document.

HizliArama function helps to perform searches of a phrase on the bookmarks of PDF. First parameter refers to where PDF locates in the computer while second parameter shows the phrase which will be searched in bookmarks.

Raporlama function provides viewing the results, which acquired after the query process carried out on any database table as a PDF file on a web browser (Internet Explorer, Firefox, Chrome, etc.). The function takes the result sets of query as a parameter.

Thanks to this plug-in, programmers are easily able to fulfil these processes without learning complex structure of PDF. It will be sufficient to include "PDFMaster.jar" in their applications.

2.1.1. Searching text inside PDF file

The search operation inside PDF document can be divided into two parts as the "Arama" function, which makes searching content of the document, and the "HizliArama" function, which makes searching on the document's bookmarks.

The search process in the content of PDF document (advanced search) has an advantage as compared to search process made on bookmarks to obtain more accurately the searched data. However, process of converting the content of PDF document into a plain text requires too much time, so advanced search process is at a disadvantage in terms of speed.

The results acquired from search on bookmarks (fast search) aren't as successful as search in the content of the document. However, fast search process provides a great advantage in terms of speed since it is only bookmarks that will be converted into a plain text.

We stated above that PDF documents are in binary file structure. As a result of this, data in document content aren't directly readable. PDFBox helps to convert the content of PDF document into a plain text.

In advanced search process, conversion of PDF document’s contents to plain text is done in groups of ten pages instead of whole document. The conversion and search processes are performed separately on these groups. If phrase is found after search, the first page number of the group where the phrase is found returns.

In fast search process, iText library is used for getting the bookmarks of PDF document. Bookmarks resemble a tree; you can perform search process after you have passed all nodes. If searched phrase has been found in bookmarks, the page number pointed to by bookmark in which phrase located is returned back.

2.1.2. Creating PDF document

We have already mentioned that the Raporlama function of the plug-in accommodates it to be displayed query results as PDF document. For this reason, stages of creating a PDF document should be examined.

iText library has been used in order to create a PDF document. The document creation could be examined in five stages.

2.1.2.1. Creating a document

You can determine the size, background color and margins of the first page with iText library while creating a document object, if you wish (Lowagie, 2007).

2.1.2.2. Determination of document output unit

Document output unit may be a PDF document, rich text file having the same content, Java servlet output stream or a web page in HTML format. Therefore, the document type should be determined first.

2.1.2.3. Opening the document

In the documents which have been created without parameters; such properties as size, background color and margins of the first page have to be specified before the document is opened. The first page of document whose properties have not been specified are opened in default settings. In such documents, the programmer can determine page properties only after from the second page. The reason of why the whole document isn't edited is that the programmer can modify every page with different properties like sizes, color, etc.

Many documents include version information and general information in the file header. For this reason, version information and metadata must be specified before the document is opened (Adobe Systems Incorporated, 2005). A PDF file which is opened without determining the version is created as default in 1.4 version (Lowagie, 2007). Before the document is opened, the version of the PDF document must be defined.

The metadata of PDF document is kept in information catalogue. Metadata must be determined before PDF document is opened. Creator or the name of the library of PDF document could be changed even after document is produced (Steward, 2004). However, it is not ethically right. Consequently, metadata of the PDF document is completed, and it is opened. Now you can add content to the document.

2.1.2.4. Adding content to the document

Any content can be added to the PDF document with three different methods:

The easiest method of adding content is to utilize simple blocks of iText library. You can use the Paragraph() function of iText in order to add content to the document (Lowagie, 2007). Thanks to this function, it isn't required to specify the parameters such as margins or where the content located in the document.

If you are an expert at PDF, you can use the other method, which is iText functions corresponding to PDF operators (Lowagie, 2007).

The last method is to use iText classes and Graphics2D functions in java.awt class if you are an expert at Java programming language (Knudsen, 1999).

2.1.2.5. Closing the document

Until this stage, all operations are saved in memory. PDF document has to be closed so that it can be created on hard disk.

2.1.3. Reporting

Reporting has a significant importance in order to archive data which has been acquired or produced. Therefore, data which have been produced with the help of reporting could be shared with the users by using various methods. It is better not to forget that users are free to use many

operating systems and platforms. For this reason, the document which will be shared must be viewable in every platform. PDF as a file format which can be viewed in every platform may be usable for reporting.

One of reporting fields commonly used in computer technology is query results acquired from database tables. Such query results may be produced in PDF and easily shared with users.

A function which takes query result obtained from table as a parameter and reports the data on web browser as PDF document has been developed in this plug-in. Thus, the programmer who is going to use the plug-in doesn't need to know complex structure of PDF for reporting. It is sufficient to give the query result as a parameter to the function.

The rs variable from the ResultSet class that is sent as a parameter to the reporting function contains the names and data of columns acquired by query performed. Reporting is performed in PDF format with iText by obtaining data according to column names respectively on each row of query results.

2.1.4. Displaying specific part of PDF document

We already stated that displaying a part of the PDF document is one of the program's primary objectives. Visualization process includes ten pages starting from page number which have returned after search process.

PDF file is transferred to the user's web browser directly from the memory. In this way, it is unnecessary to create PDF document parts, which each one consists of ten-page block, on hard disk during the advanced search process. This structure eliminates such problems as the deletion of PDF part groups, which are based on query result performed on the documents, by the users from hard disk and when the deletion will occur. The programmer doesn't need to deal with problems thus.

Viewing process is performed with Servlet instead of JSP. Displaying PDF in the web browser by using JSP may lead to numerous problems. This is because each web browser (Internet Explorer, Firefox, Chrome, etc.) interpret HTML codes differently from each other.

Some servers don't recognize JSP output as binary. Therefore, you see question marks in the content. PDF files are registered on the server's file system without any problem; however, the user will see a blank page after opening the PDF files on an available link. When you attempt to view a PDF file with JSP, you will have to add indentations, spaces, etc. If you desire to create a binary content, it won't be appropriate to be written with JSP codes. The best way to view PDF documents is to use Servlet structure.

The user must have a PDF reader or have a suitable PDF reader plug-in of the web browser in the computer so that the document can be viewed. If the user installs Adobe Reader, a PDF viewer program produced by Adobe corporation, in his/her computer, PDF plug-in will have been set up to web browser as well.

3. Results

One of our major goals, such as displaying a part of PDF document, has been accomplished and so copyright of the document has been protected. We have stated above that the search function in the content of PDF document referred to as advanced search gives more accurate results as compared to search process made in page bookmarks referred to as fast search to obtain the

data which are searched. However, advanced search isn't fast enough as compared to fast search because it takes much time for conversion. In addition, the reporting function in the electronic library plug-in will provide programmers with great convenience in reporting their data as PDF documents.

As a result of studies on the plug-in developed, it has been concluded that phrase search performed by separating PDF document into the parts is very advantage in terms of speed as compared the phrase search performed on whole PDF document. The reason of this is that the other parts are skipped when the searched phrase is found. Therefore, if the searched phrase is how close to beginning of file, so much time is gained. In addition, any error conditions which cause buffer overflow during search process for big PDF documents have also been avoided.

In real-time search operations, various studies have been carried out on the PDFs given in Table 1 for analysis of the factors that can affect the speed of search performance. PDF documents in Table 1 have been created as special according to its properties such as file size, bookmark count, page count and content type.

Each of the PDF documents in Table 1 is placed to the search location one by one. After that, the performance tests are carried out in a way the exactly same searched phrase in the 30 page blocks of the relevant PDF document can be found. It should be noted that there is only one PDF document in the search location during performance tests. Document properties have been determined to allow to be made inferences by making comparisons between PDF documents.

The main purpose of performance tests is to observe the effect of each document property on search performance thanks to carefully created documents instead of a system performance that will include all PDFs. The results observed from the test operations are shown in Table 2 for advanced search and Table 3 for fast search.

All values in the tables were obtained in milliseconds on the Intel i7-3630QM processor. CPU usage should be minimized for the best values.

We know that the content of PDF document is in the binary file structure and must be converted. However, as you can see from the Table 2 (PDF2, PDF3 or PDF6) images, figures and blank pages in the PDF document are skipped during conversion process. As a result of this, there is no direct effect of these structures on search performance, but they increase the file size of document. Depending on values in Table 2, it has been determined that as the file size (PDF3 vs PDF6) or page count of the PDF document (PDF1 vs PDF4) increases, the search speed and hence performance degrades. However, it can be easily realized that the effect of the file size on the search performance is more than the effect of the page number of the document.

The search for bookmarks is directly related to the size of the document, but the effects of the library functions used to obtain the bookmarks of the document affect the search performance. While retrieving bookmarks, it has been determined that the iText library was almost four times faster than the PDFBox library, so the iText library has been preferred for the fast search. The results in Table 3 is obtain with the iText library function. Indirectly, the search speed can vary depending on the size of the file (PDF3 vs PDF6), the page number (PDF1 vs PDF4) and content of the file (PDF5 vs PDF6). There is nearly no effect (PDF2 vs PDF3) of bookmark count on the search performance.

Table 1. The various PDFs (PDF1, PDF2, PDF3, PDF4, PDF5, PDF6, PDF7) and their properties

PDFs	File Size	Bookmark Count	Page Count	Content Type
PDF1	11.7 MB	7	243	0-210 Only Text, 211-243 Only Image
PDF2	11.7 MB	39	305	0-90 Only Text, 91-120 Blank Pages, 121-150 Text-Image Mixed, 151-180 Only Image, 181-305 Only Text
PDF3	11.7 MB	7	305	0-90 Only Text, 91-120 Blank Pages, 121-150 Text-Image Mixed, 151-180 Only Image, 181-305 Only Text
PDF4	11.7 MB	7	488	0-460 Only Text, 461-488 Only Image
PDF5	1.75 MB	7	305	0-305 Only Text
PDF6	4.6 MB	7	305	0-90 Only Text, 91-120 Blank Pages, 121-150 Text-Image Mixed, 151-180 Only Image, 181-305 Only Text
PDF7	11.7 MB	7	305	0-276 Only Text, 277-305 Only Image

Table 2. The performance table in milliseconds for advanced search function on various PDFs

PDFs \ Page Count	30	60	90	120	150	180	210
PDF1	786,85	1128,04	1488,58	1834,31	2180,51	2497,7	2843,8
PDF2	805,18	1162,6	1507,52	1514,9	1699,45	1708,54	2056,15
PDF3	814,06	1157,61	1514,47	1518,09	1704,48	1707,93	2035,35
PDF4	871,39	1219,55	1574,43	1913,57	2252,5	2594,12	2940,58
PDF5	511,46	836,9	1179,56	1515,34	1858,25	2198,41	2551,25
PDF6	562,78	897,57	1239,27	1241,65	1410,66	1413,22	1747,43
PDF7	803,31	1163,62	1512,17	1861,98	2169,69	2510,28	2847,78

Table 3. The performance table in milliseconds for fast search function on various PDFs

PDFs \ Page Count	30	60	90	120	150	180	210
PDF1	104,59	104,68	104,82	103,88	104,66	105,4	104,67
PDF2	132,62	132,64	132,8	132,59	133,39	132,8	133,02
PDF3	127,13	127,27	126,76	127,39	126,37	126,75	127,17
PDF4	195,68	195,73	196,19	195,82	195,37	195,2	195,74
PDF5	128,64	129,74	130,54	130	129,31	129,86	130,44
PDF6	98,88	98,62	98,66	98,85	98,77	98,8	98,83
PDF7	130,87	130,85	131,09	130,81	130,9	130,93	131,36

4. Discussion and Conclusion

The protection of rights belonging copyrighted publications, which is the main purpose of the study, has been successfully achieved.

Based on the information given in the study, an application framework which, like digital libraries with index specified in introduction section, can be developed that performs the conversion process as bulk on a PDF set, then searches only on plain text. In addition, the process described in the section of displaying a specific part of the PDF document is included as a function in the plug-in. Since the plugin is open source, developers can take advantage of the related function and separately define the part displayed for paid publications in accordance with publisher policies. Performance of the framework can approach ideal values with the help of parallel programming technique and suitable hardware platform.

It has been stated above that we have developed two different search methods for users, and advanced search is much more effective in reaching the desired data but slower than fast search. In order to further examine the performance of the search functions, some studies have been carried out on documents created as special according to parameters such as page number, file size, content and bookmark count.

In advanced search, it can be assumed that the value obtained for the last block of the PDF document is almost equivalent to the time required to convert the PDF document to plain text. In addition, in Table 2, parameters such as file size, page count used for advanced search can be obtained with the help of iText library without the need to convert the document to plain text. Based on this information, the time required for organizations that have bulk electronic documents to become an indexed digital library can be roughly calculated with an application to be written according to the related parameters whose effects can be detected from Table 2. Thus, obtaining preliminary information about the process will be useful in many ways.

Users can develop special applications according to their wishes and various PDF tools by using the PDFMaster plug-in.

References

McCray, A. T. and Gallagher, M. E., 2001. “Principles for digital library development”, Communications of The ACM, 44, 48–54.

Kumar, R. and Pamula, R., 2020. “Social Book Search: a survey”, Artificial Intelligence Review, 53, 95-139.

Na, N., 2007. “Testing the Boundaries of Copyright Protection: The Google Books Library Project and the Fair Use Doctrine”, Cornell Journal of Law and Public Policy, 16, 417-448.

Illyes, G., “PDFs in Google search results”. https://developers.google.com/search/blog/2011/09/pdfs-in-google-search-results (12.01.2021).

Hasbrouck, E., “What is the Internet Archive doing with our books?”. https://nwu.org/what-is-the-internet-archive-doing-with-our-books (13.01.2021).

Christenson, H., 2011. "HathiTrust: A Research Library at Web Scale", Library Resources & Technical Services, 55, 93-102.

Warnock, J., “The Camelot Project”. https://planetpdf.com/planetpdf/pdfs/warnock_camelot.pdf (05.02.2020).

Atalar, C., “Electronic Library”. http://www.electroniclibrary.ml (25.07.2020).

Apache Software Foundation, “Apache PDFBox”. https://pdfbox.apache.org (24.12.2019).

Lowagie, B., 2007. “iText in Action Creating and Manipulating PDF”, Manning Publications Co.

Atalar, C., “Electronic Library Plug-in called PDFMaster”, https://github.com/celalatalar/pdfmaster (23.01.2021).

Adobe Systems Incorporated, 2005. “Acrobat 7.0 PDF Open Parameters”, Addison Wesley.

Steward, S., 2004. “PDF Hacks”, O'Reilly Media.

Knudsen, J., 1999. ”Java 2D Graphics”, O'Reilly Media.