Volume 2, No. 8, August 2011
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

EXTRACTING PLAIN TEXT FROM CORRUPTED WORD DOCUMENT

Saptarshi Naskar[1], Souvik Sarkar[2] and Krishnendu Basuli[3]
[1]Department of Computer Science, Sarsuna College, INDIA
sapgrin@gmail.com
[2]Department of Computer Science, RKMRC, Narendrapur, India
[3] Department of Computer Science, WBSU, India

INTRODUCTION

Text conversion is a process written in some programming language whose main task is to extract the plain text from the supplied source file in some format, and put the text in the file of different format. In this conversion procedure the extension of the file will be changed but the data within it remains unchanged in format and in size of the actual data. This conversion procedure is done through some programming language (may be written in C or C++ or some other programming language), whose main task is to read the source file line by line (in many systems it reads per character) and whenever the appropriate text (i.e. the valid data or character) is found it then copies the entire text into a different file format, remaining unchanged the format of the text that is in the source file.

Here, we design the text converter in C programming language which accepts the file format with an extension of *.doc*, *.rtf* and also *.txt*, and extracts the plain text from these files and put the texts in a *text file* with remaining unchanged of the text format of the source file.

WORKING PROCEDURE

Actually, when we supplied a filename along with its extension and the full path of the source file, the program written in C language check firsts what type of extension have the source file. It then call the appropriate function for extract the plain text from the source file and put it in the text file and save it in appropriate location.

The program of Text Converter, written in C language is responsible for opening the *doc* file format written in *MS Word 2000* or in *MS Word 2003*, extracts the plain text from these files and put it in the text file.

It also responsible for opening the *rtf* file format written in *MS Word 2000* or in *MS Word 2003* or in simple *WordPad* format, extracts the plain text from these files and put it in the text file. Besides on these, the extra feature is that it can also open the *text* file written in notepad format.

SCOPE

It is meant for us by the developer and will be the final basis for validation of the final delivered system. This document is final and it can't be altered once it is acknowledged by the customer.

DEVELOPER'S RESPONSIBILITIES

The developer's responsibilities are:-

a) developing the system
b) installing the software on the client's machine
c) providing any user training that might be needed
d) maintaining the software for a specific time

PRODUCT FUNCTIONS OVERVIEW

This software provides a basic conversion of any text files from that of MS WORD, WORDPAD formats to a plaintext i.e. notepad format. The software additionally provides an encryption and decryption function on all plaintext i.e. notepad format documents.

USER CHARACTERISTICS

The main users of this system will be computer operators having a sound knowledge of using softwares such as editors, word processors etc. And has a basic knowledge of computer fundamentals.

ALGORITHM FOR TEXT CONVERTER

In the following algorithm of text conversion the main function accepts two parameters, first one is the exe file and second one the source filename along with its path and extension. It then checks the extension of the supplied file format, if it is a valid one, then the corresponding functions will be invoked to extracts the plain text from it.

Algorithm (CONVERTER):

Step1: Call of main function with two arguments

(a) Char *argv[ ] /*For passing file path as command line arguments*/

(b) Int argc /* Counting no of arguments have passed*/

Step 2: Declare input & output file pointers *ip,*op. /*initialize variables*/
Declare integer variables b, w, character=0,i=0,a=1.
Declare character variable as c and character pointer variables *x, *v.
Define the string constants to the character pointers as *d ="doc",*t ="txt",*r ="rtf".

Step 3: using a while loop we determine the extension of the supplied
file have-
repeat while until *x!='\0'
if(*x=='.') then
copies the extension of the source file in *v.

Step 4: if(*v==*d) /*supplied file is a .doc file*/
Open the file in argv[1] in read mode.
Open the file c:\doc.txt in write mode.

Step 5: while a<=6*80+13 /*discarding extra characters*/
Reads each character and file pointer increment by one.

Step 6: read a single character after discarding the previous characters
And check the ASCII value of it.

Step 7: if (b==55) then /*the supplied file format is written In MS Word 2000*/
while (a<=43+12*80+16) do /*discarding extra characters*/
fscanf(ip,"%c",&c);
a++; /*end of while*/

Step 8: use a while loop to read each character from the file and do the following operation until End Of File has not been reached assign the value of each character in integer variable b and does the following operations using if-else structure

Step 9: if-else structure starts by checking condition whether b=13
If b==13 then
the output pointer prints 0x0d
the output pointer prints 0x0a
/*End of if structure*/
Else
file pointer writes every character in destination file and points to next character /*end of while*/

Step 10: close the output and input file pointer. /*end of inner if*/

Step 11: else /*the supplied file format is written In MS Word 2003*/
while (a<=43+25*80) do /*discarding extra characters*/
fscanf(ip,"%c",&c);
a++; /*end of while*/

Step 12: use a while loop to read each character from the file and do the following operation until End Of File has not been reached assign the value of each character in integer variable b and does the following operations using if-else structure

Step 13: if-else structure starts by checking condition whether b=13
If b==13 then
the output pointer prints 0x0d
the output pointer prints 0x0a
/*End of if structure*/
Else
file pointer writes every character in destination file and points to next character /*end of while*/

Step 14: close the output and input file pointer. /*end of outer if*/

Step 15: else if(*v==*t) /*supplied file is a .txt file*/
Open the file in argv[1] in read mode.
Open the file c:\text.txt in write mode.

Step 16: use a while loop to read each character from the file and do the following operation until End Of File has not been reached assign the value of each character in integer variable b and does the following operations using if-else structure

Step 17: if-else structure starts by checking condition whether b=13
If b==13 then
the output pointer prints 0x0d
the output pointer prints 0x0a
/*End of if structure*/
Else
file pointer writes every character in destination file and points to next character /*end of while*/

Step 18: close the output and input file pointer. /*end of outer if*/

Step 19: else if(*v==*r) /*supplied file is a .rtf file*/
Open the file in argv[1] in read mode.
Open the file c:\rtf.txt in write mode.

Step 20: using a for loop do the following
for(i=0;i<8;i++) do /*discarding extra characters*/
fscanf(ip,"%c",&c); /*end of for*/
fscanf(ip,"%c",&c); /*read the next character and
b=c; check the ASCII value of it*/

Step 21: if(b==100) /*opening rtf file format written in MS Word 2003*/
using a for loop do the following
for(i=0,j=1;i<j;i++,j++) do /*get the valid format*/
read a single character and assigned the ascii value to b
if(b==92)
read a single character and assigned the ascii value to b
if(b==105)
read a single character and assigned the ascii value to b
if(b==110)
read a single character and assigned the ascii value to b
if(b==115)
read a single character and assigned the ascii value to b
if(b==114)
for(k=0;k<11;k++)
read a single character and assigned the ascii value to b
if(b==32) then
k=11;
/*end of inner if*/

Step 22: use a while loop to read each character from the file and do the following operation until End Of File has not been reached assign the value of each character in integer

variable b and does the following operations using if-else structure

Step 23: if-else structure starts by checking condition whether b=13
If (b==13||b==123) then
the output pointer prints 0x0d
the output pointer prints 0x0a
/*End of if structure*/
Else if(b==92)
for(k=0;k<3;k++) do /*discarding extra characters*/
fscanf(ip,"%c",&c); /*end of for*/
k=0; /* end of if*/
Else
file pointer writes every character in destination file and points to next character /*end of while*/
/*end of for loop*/

Step 24: close the output and input file pointer. /*end of inner if*/

Step 25: else /*rtf format is written between WordPad and MS Word 2000*/
for(k=0;k<15;k++) do /*discarding extra characters*/
fscanf(ip,"%c",&c) /*end of for loop*/

Step 26: if(b==100) /*rtf file format written in wordpad*/
while (a<55+37) do /*discarding extra characters*/
fscanf(ip,"%c",&c);
a++; /*end of while*/

Step 27: if-else structure starts by checking condition whether b=13
If (b==13||b==123) then
the output pointer prints 0x0d
the output pointer prints 0x0a
/*End of if structure*/
Else if(b==92)
for(k=0;k<3;k++) do /*discarding extra characters*/
fscanf(ip,"%c",&c); /*end of for*/
k=0; /* end of if*/
Else
file pointer writes every character in destination file
and points to next character /*end of while*/
/*end of for loop*/

Step 28: close the output and input file pointer. /*end of inner if*/

Step 29: else /*rtf format is written in MS Word 2000*/
while (a<55+30*80+8) do /*discarding extra characters*/
fscanf(ip,"%c",&c);
a++; /*end of while*/

Step 30: if-else structure starts by checking condition whether b=13
If (b==13||b==123) then
the output pointer prints 0x0d
the output pointer prints 0x0a
End of if structure/
Else if(b==92)
for(k=0;k<3;k++) do /*discarding extra characters*/
fscanf(ip,"%c",&c); /*end of for*/
k=0; /* end of if*/
Else
file pointer writes every character in destination file and points to next character /*end of while*/
/*end of for loop*/

Step 31: close the output and input file pointer. /*end of inner if*/
/*end of outer if*/

Step 32: print Wrong Argument /*supplied file format is Exit from the main neither doc nor txt nor rtf*/

Step33: stop.

CONCLUSION

The TEXT CONVERTER is a software using which one can easily convert the text files of .doc i.e. MICROSOFT WORD format, .rtf i.e. WORDPAD format to plaintext i.e. notepad format for easy viewing, less memory space utilization and readily accessible ways.

REFERENCE

[1]. Microprocessor and Interfacing Programming and Hardware, Douglas V. Hall
[2]. Using Assemble Language, Allen L. Wyatt
[3]. Network Programming in C, Barry Nance

