Production Project
SimplePDF

by Pamela P. Cote

April 3, 2002

For Electronic Coaching Systems

Professor Robert Krull

Introduction

The application reviewed in this project is of value to two segments of my clientele. One client market includes independent consultants who write papers for professional publications and to publish on their websites. The other is an HR consultant who needs to publish standard forms and policy on intranets.

According to Preston Gralla, Executive Editor, ZDNet Software Library, Friday, October 19, 2001:

"…no matter what kind of computer you have, what software you have, or what browser you use, you can read Acrobat files on the Web--or anywhere else, for that matter. All you'll need is a free copy of Acrobat Reader."

Since the hefty list price of $249 for Adobe Acrobat was beyond my reach. I did a quick visit to ebay.com and found the current version selling for over $180, and the previous version selling for $90. I then searched ZDNet.com and found Simple PDF for $35.

On the website simplepdf.tangentially.com, the product promises to:

> *"Move the design review process beyond the walls of your organization. Save valuable time, and eliminate the cost of handout deliveries with **simplePDF**. **simplePDF** offers you a scalable, secure collaborative workflow solution for creating and sharing pdf documents online within a group, and getting feedback . With its easy-to-use interface, facility to incorporate multiple page sizes, and capability of embedding multimedia files, it is virtually a Desktop Publishing (DTP) system."*

In addition, It's audience is stated to be in a

> *"… collaborative work situation…*
> - *students submitting project work to their teachers*
> - *colleagues at a corporate workplace preparing a presentation report together*
> - *legal eagles developing and refining legal agreements between several signatories*
> - *designing professionals evaluating alternative compositions before hitting upon that perfect solution*
> - *administrators looking to generating online policy documents from multiple sources "*

A summary of my experience illustrates why I decided it was a perfect example for my ECS project.

I downloaded a trial version, installed and launched it, and my confusion began when I tried to create my PDF. A Quick Start Guide was displayed and I followed the first step: Add page. I had to close the Quick Start Guide in order to follow the steps. I tried to get help information by clicking on the ? and got the Quick Start Guide. After completing each step, I had to restart the Quick Start Guide, determine if I performed the action correctly, read the next step, and close it to proceed.

The file types indicate that I can only use .txt or .jpg files. I tried a Word document and a version in RTF. The RTF appeared to load, but all the coding was included. The Word document elicited an error message, but no explanation. I saved the document as a .txt and got the same error message. I opened the file in WordPad and saved it again to the same name. I was then able to load it as a snippet.

The text didn't wrap, so I looked for a method to edit it. The icon with a pencil on paper at the bottom of the snippet looked promising, so I hovered over the area. The tool tips were helpful to get to the edit screen.

I enlarged the font, and then wanted to preview what I had. I found no preview available, so clicked create PDF. This opened the document using Adobe Acrobat Reader. The text was truncated, not wrapped.

At this point, my impression of the product was - simple as in doesn't do much, not simple as in easy to use.

When I printed the user manual, I found a section on page 5 (of an 8 page document) that states: "any text that falls outside of the snippet area will be truncated". I opened the file again in Wordpad, wrapped text to ruler and saved the file. I deleted the page in composer, started over with a new snippet with the new txt file. It still didn't wrap the text, and I think it shouldn't be this much trouble to accomplish my goal. Later when compiling topical content help, I found a reference that says the stream editor is not WYSIWYG.

The Product's Existing Help System

1. Getting Started Guide
   The product came with a quick start guide (figure 3), giving step by step instructions to create a PDF file.

   The user can select whether this guide appears when the program starts. It can also be viewed by clicking on the  button. Unfortunately, you can't do anything while this getting started guide is displayed. You have to close it. Otherwise all you get are "doink" sounds.

2. User Manual
   The product came with a PDF manual named composer.pdf which could be printed after opening it from the getting started guide. It was titled "simple-PDF, making life a little easier…" I found this to be more of a marketing tool than user manual.

   Nestled at the bottom of page 5 is the comment "Remember that PDF would truncate any text that falls outside the snippet area. As the current version of Simple-PDF does not have text metrics capability, it may inadvertently crop your text." That seems like something I should have known up front.

   I also found a second document with better information while looking through the directory. This document was named simplepdf.pdf. Terms were explained better and I think the process was easier to understand.

3. Tool Tips
   There are tool-tips which the user can turn off.

4. Technical Assistance
   After clicking the About button, the web site and email address are given, presumably for technical assistance. In addition, there is a Support/Relationships button, which displays a message that the "f-ticket CRM system" isn't installed. It's

possible that a registered version would have this installed and that it would be an adequate technical assistance solution.

5. Confusing Elements
   a. Poor naming
      Interestingly, there are four different buttons for exiting. Maybe they have something to do the evolution of the product and some parts of the implementation are left over from non-windows versions.

   b. Poor descriptions of process
      I wanted to know why I needed a snippet. A general review of the process of creating PDF's would have been helpful. Context sensitive help in the form of a glossary as mentioned in Horton 13 would help. For snippets, it could say: snippets are the building blocks of a PDF page. All text and images must be linked to a snippet which describes the dimensions of the text or image block. Snippets are placed on a page to create the page layout.

Target Users

My assessment of the target audience is that they are advanced users of word processing, windows and the internet. They have used PDF files in the past as downloadable forms or reference information. They may have had desktop publishing experience. These are motivated individuals who have the specific goal of turning existing or future text, image, and possible multimedia projects into PDF files. They can visualize the final product, but may have no idea how to get there. I believe that the target audience, as were my testers, is willing to try a product if it didn't cost them more than a little time. They are looking for a tool to get a specific job done.

My Redesign Of The Help System

I will discuss redesign as though it were in my power to make the changes. The most cost effective items are listed below.

Improvements

1. affordance in icons, buttons, and naming
2. gestalt principals of grouping
3. availability of Quick Start Guide when needed
4. enhancement of Quick Start Guide
5. expanded tool tips

Additions

1. graphical diagram illustrating essential concepts
2. context sensitive help via 
3. on-line topical manual

Nice to Have, but possibly cost prohibitive is a searchable index.

Affordance, Naming and Imaging

Some of the icons gave no clue as to their function. I changed the icons to fit with users existing idioms whenever possible to reduce visual processing (Coe 3 and 6). I tried to

make them easy to associate with a process since mental images and memory cues in context are easier to remember (Coe 4).

On the workspace (figure 1) the button called Support/Relationships produced a message that suggested it was a CRM system, however, the link didn't exist and I couldn't find any information related to the f-ticket system it referenced on their website. I changed the button name to **Customer Support** which is my interpretation of it's meaning.

Other problems I identified on the workspace (figure 1) include an unusual combination of buttons  in the upper right corner. The leftmost button looks like a floppy disk and would seem to mean save the file. However, after several attempts to determine its action, I found that it duplicates the upper left icon if right clicked and when simply clicked, adds an instance of the program to the Task Bar. I decided the button was confusing and removed it. Similarly, there was another icon that had no apparent meaning  (figure 1) that could not be inferred from its placement among the help system buttons. The tool-tip said "Quit Simple-PDF Composer" and it did close down the application. I felt this was redundant as well as confusing and removed it.

The **Stream Editor** (figure 5) was difficult for me to understand. For instance, since the only thing you can edit is text, why wasn't it called the Text Editor? As I came to be more familiar with the process of PDF creation, I realized that there could also be commands added and edited here. The first thing I changed were the icons for find and replace. It was easy to replace the pointing finger  with the binoculars for the find button, but I couldn't find a suitable replacement for the find and replace button. I couldn't justify a finger that looked to be turning a page  as the icon for replacing text, so simply used the word "replace". The action for the open book eluded me until I actually tried it. It turns out to be a great utility that opens a document and allows you to mark passages for copying and when you close the document, the text is inserted into the stream. This was very quick and efficient. I settled for a combination of the standard open file and paste icons  . If the icon doesn't convey the meaning well on first use, it is a good picture reminder of the action on subsequent use.

One of the icons on the Stream Editor was a green highlighted triangle () similar to what you could expect to mean more options or move to the right. Instead it meant "**save stream**". My testers had no suggestions, and I couldn't come up with a better icon that would convey the meaning. I decided to keep this icon, but add text and make the button bigger. This was to reduce mental load for the beginner or infrequent user. There was space available, and it was easy to implement from a developer view. I used the same reasoning for the PDF tools button. Although the icon looks like a set of tools on close inspection, if you hadn't dealt with this type of icon previously, it wouldn't make much sense. The tools are really too small to figure it out with a glance.

On the Quick Start Guide (figure 3), there was a button labeled "Read composer.pdf for more help". The action associated with this brought up a user manual in PDF format which could be printed. I changed the label to "**Read/print user manual**". I thought the new label was a better description of the action. After user testing, I decided to move this option to the Help screen (figure 7). I also changed the file referenced to the one named simplepdf.pdf because I found it was closer to an expected user manual than the composer.pdf marketing tool.

Grouping

In general, I found that grouping of buttons and activities on the workspace acceptable, but I changed the order to reflect the order in actual usage (page, snippet/files, sim, and PDF with more space and separators between. The snippets area takes up a large portion which is consistent with it's high frequency of use and it's easily locatable. All changes to grouping were to provide clear road maps for beginners (Coe 3), and mental mapping for intermediate and advanced users. In addition, they reflect proximity, continuity, and common fate principles (Coe 2)

I kept all the **Help** buttons in the same area and redefined their meanings.

I added task oriented headings to the Quick Start Guide (figure 4) to increase the ability to quickly return to the correct step in the process.

I added a new Help Screen discussed later and added buttons for other help elements to this screen.

Tutorial – Quick Start Guide

The only tutorial is the "Quick Start Guide" which pops up the first time the product is used and every time the application is opened unless the correct box is unchecked. However, this guide didn't remain open during execution of the instructions. For instance, you got a "doink" if you tried to click on the Add Pages button when following step 1. I changed the mode to make this guide viewable during activity on the workspace.

After testing, I decided this element should be launched from the Help screen instead of the workspace because the usefulness would quickly become obsolete. I maintained the original implementation of this screen popping up during the first use of the product. This way the user quickly sees what needs to be done and in what order.

I added task oriented headings to quickly return to the correct step in the process (Krull, Basic Imaging Concepts).

I added a diagram of the main concepts of the simplePDF schema. This helps the user visualize how the elements work together to create the .sim file.

This element has a short half-life, but I believe that the marketing potential is increased if the trial user can be successful on the first try. They are impressed with the thoroughness and usability of the product. This type of Quick Start Guide encourages the user to go directly to "doing" which is their preferred method of learning (Coe 6). And they can visualize the end product by following certain steps.

New Help Screen

I created a new Help screen accessed by clicking on the Help button on the workspace. From the Help screen (figure 7), one can open the Quick Start Guide, read or print the User Manual, and look up topical information.

Context Sensitive Help

When the ? was clicked, it brought up the getting started guide. I changed it to have similar meaning to Microsoft products, in that after clicking the question mark, you can click on the workspace – snippets for example – and glossary type help similar to what Horton mentions (Horton 13) appears with a link to the contents page of the topical help.

I changed the error message I received when trying to select a word document as a snippet stream (figure 9). The file existed, but wasn't an acceptable type of file, so I added that basic information to the message (figure 10). Otherwise, a user can spend fruitless minutes trying to figure out why the file couldn't be opened when selected from the file listing.

I enhanced the Help by adding expanded basic information in conjunction with the ? and tool tips. This enhancement would take little time for someone experienced with the product to create. Adding a link to the help contents pages is also a quick method to bring help to the fingertips of those wanting more information. It would give a reason to the "Show hints" checkbox. Unchecking after graduating to the next level of experience means not having the tips pop up every time you move your mouse over the item.

Topical Contents

I created a topical contents lookup on the Help screen (figure 7). I found ample information already written that simply needs organizing and linking. This will give the first time user more confidence in the product and would be cost effective. I used "cut and paste" while searching through existing files from the simplepdf directory on the basic concepts of editor, images, page, sim files, snippet, stream, and text. I compiled an adequate amount of help text in less than 2 hours. The text needs to be organized and edited for redundancy, but would be cost effective as a long term help aid for the intermediate and advanced user.

The topical contents I created (appendix A) was very helpful to the test users. I believe it would make the product more appealing to first time users. Sales of the product depends of how effective the product is in reaching the users goal. This is a very inexpensive product, and offers no searchable help index. Since I found an abundance of existing information, I think a logical future enhancement would be to organize and add search capability.

Evaluation

Four people participated in the testing and review of my proposed changes. Their technical expertise fit my assessment of the target audience in that they were advanced users of word processing, windows, and the internet. Some had desktop experience from beginner to advanced. Some had previous experience in creating PDFs. I debriefed them via on-line questionnaires, email, and telephone. I limited my question on their profiles (Coe 8) to determine there schemata, learning curve placement, and motivation. I also realize that they can be too polite, but tried to convince them that I wanted real observations.

I was surprised that the testers of the “before enhancements” phase found the product easier to use than I did. Maybe I’m more sensitive to deficiencies of help systems, or they had a much smaller and simpler goal.

The testers of the second phase commented that the proposed enhancements made the product intuitive. One tester commented: “The help system was very nice and assisted in the use of the product.”

Some users were motivated by what the product could do for them, while others tested as a favor to me.

I thought those users with previous experience creating PDF files in the past would have a head start, but according to one regarding previous experience “It was somewhat helpful, but I don't think it made a tremendous difference. The terminology and options were easier to understand than say, Acrobat.” Another that had no previous experience said, “I actually hadn't created them in the past, which is probably why I was so intrigued by being able to do them. I've always wanted to create them - especially as forms on company intranets. Terminology seemed fine to me and the editing was ok. I only need simple documents so I haven't tried all the features.”

I asked testers if they would recommend the product with the proposed changes. One was hesitant without first defining product requirements and then comparing it to other products, and another who had a defined need for this type of product was very enthusiastic.

Summary

The product is easy to use, and people generally would use it to serve a specific purpose, thereby bringing their own motivation. They have a good idea of the end product or goal they are trying to achieve. The addition of the diagram in the help system makes visualizing the process easier. The naming of major concepts and elements is limited and while foreign to most new users, becomes quickly incorporated into their PDF paradigm/schema. Performing the steps described in the improved Quick Start Guide achieves the expected results in little time. Evaluation can only be achieved by generating a PDF and viewing it through the integrated product Acrobat Reader. The editor is not wysiwyg, but suggestions of how to overcome the line length editing are given. The file can be saved as a .sim so that additional editing can occur as needed so redoing and reevaluation can occur. The final product that matches the original goal brings closure to the process. (Krull, Components of a Coaching System). It is also is motivating to continue to use the product. The proposed help system makes it easier to succeed.

Changes to the Workspace

Figure 1, workspace before design changes

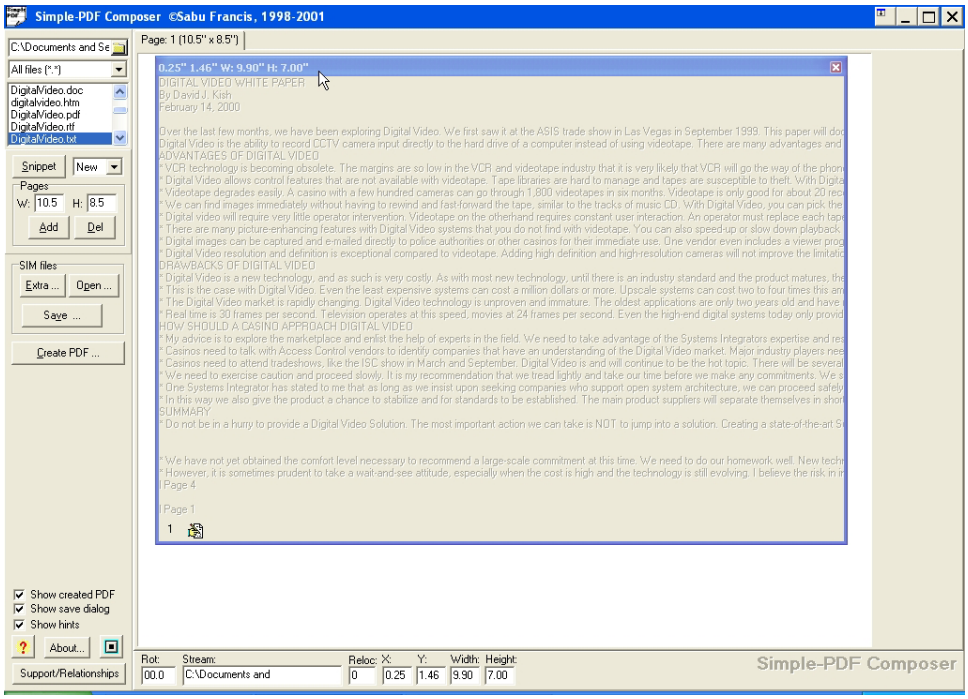


Figure 2, workspace after design changes

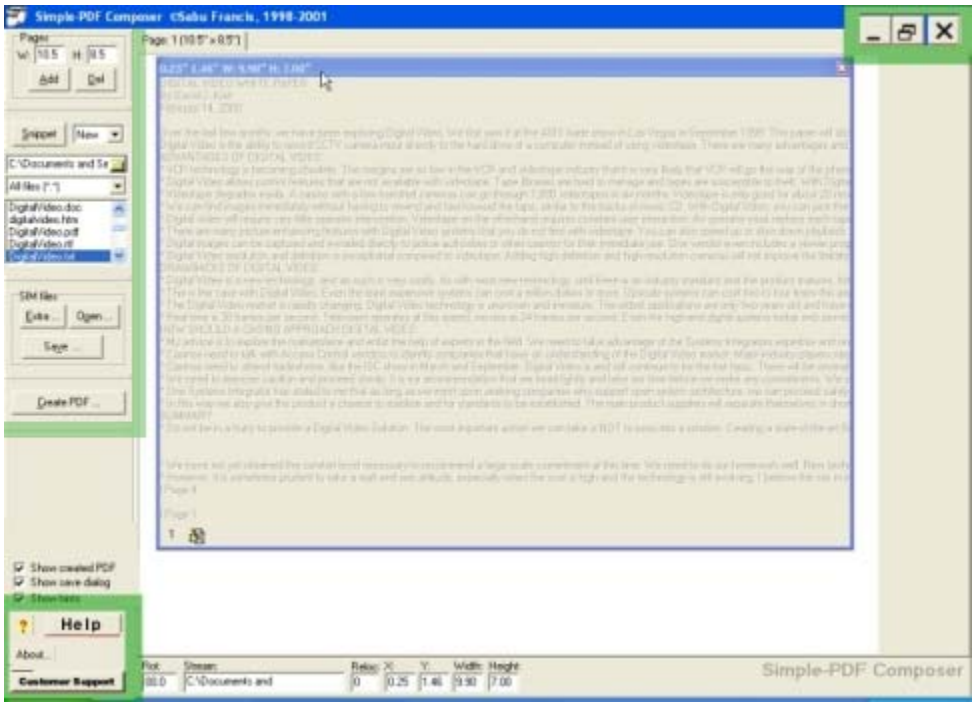


Figure 3, Quick Start Guide before design changes

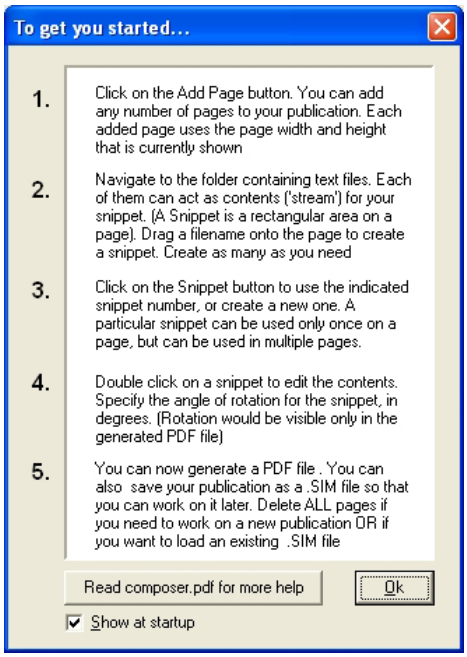
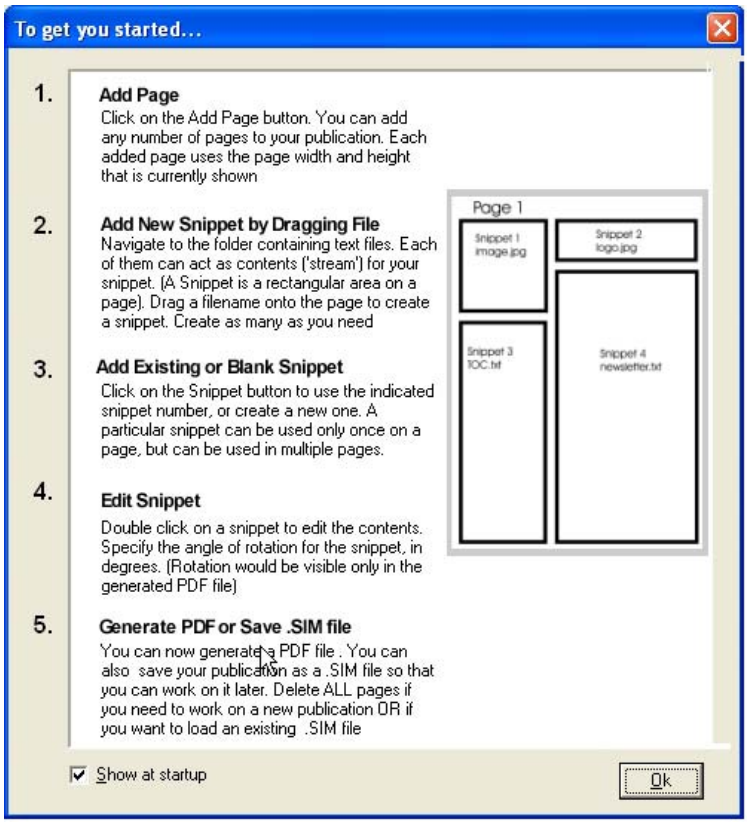


Figure 4, Quick Start Guide after design changes



Changes to the Stream Editor

Figure 5, Stream Editor before design changes

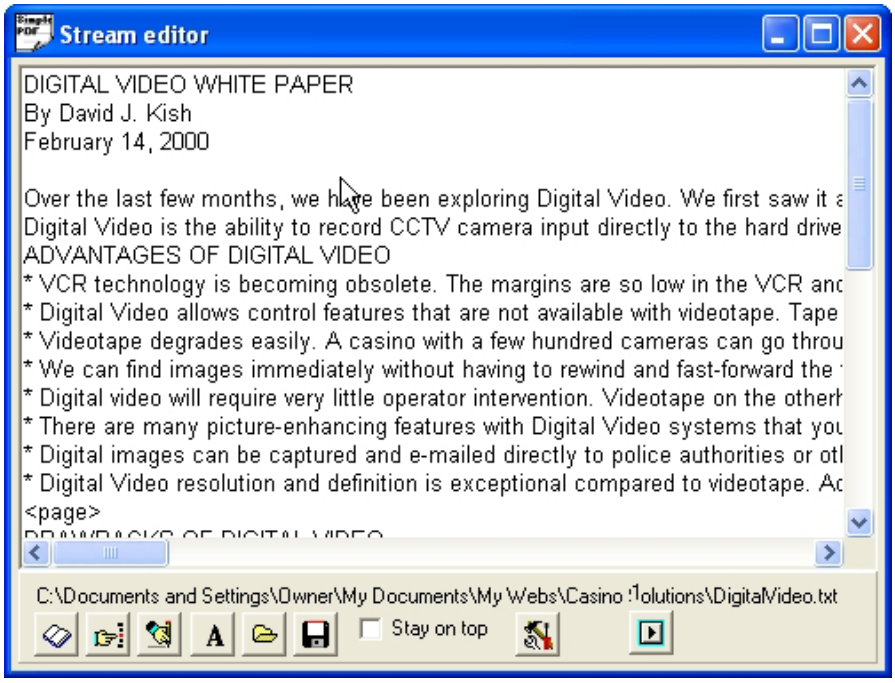
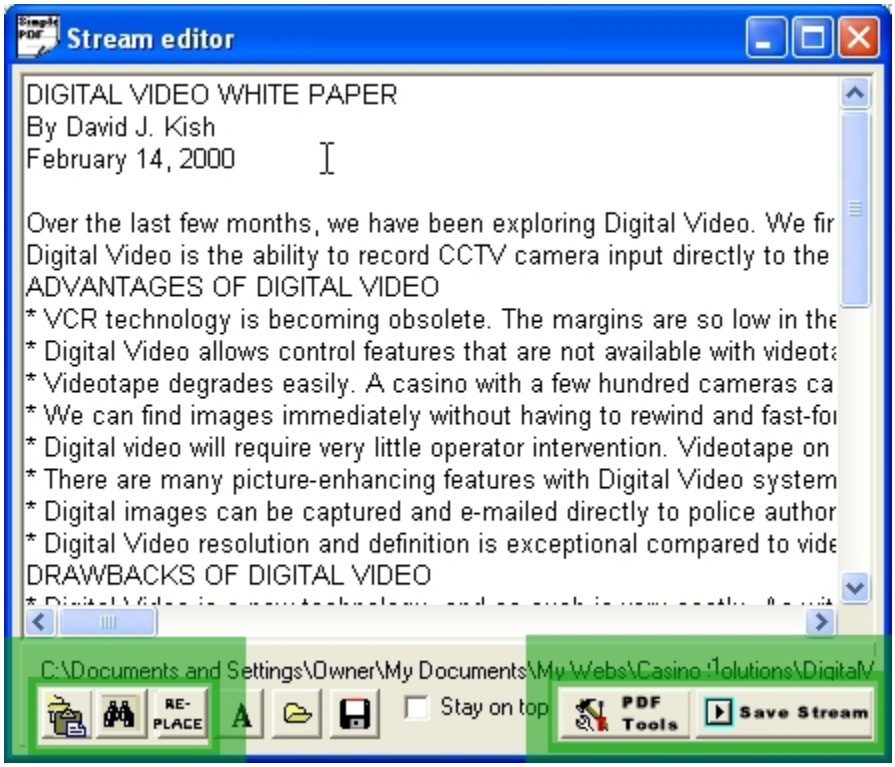


Figure 6, Stream Editor after design changes



Help Screen and Context Sensitive Help

Figure 7, New Help Screen

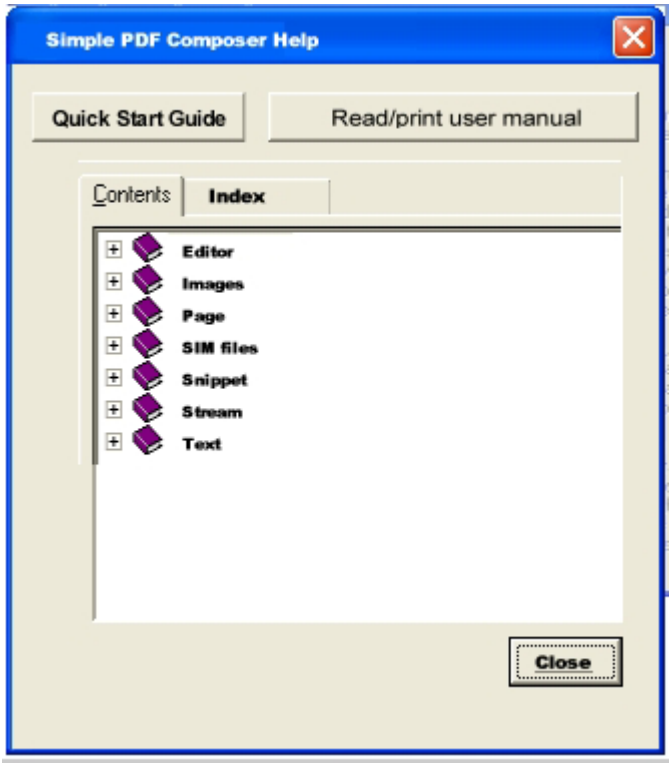
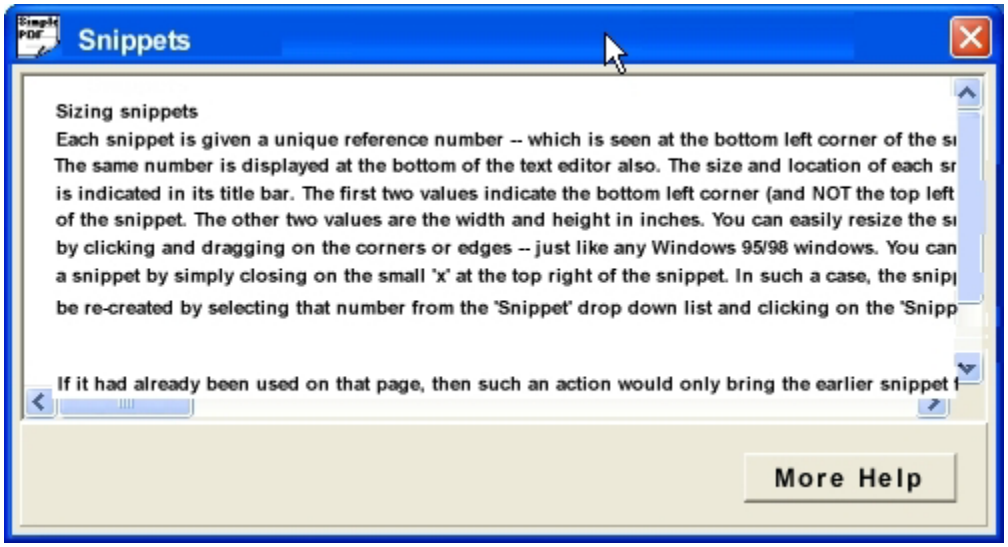


Figure 8, New context sensitive help



Error Message

Figure 9, Error message before design changes

Figure 10, Error message after design changes to add help text

Appendix A, Topical Content

(Note: This is a compilation of existing simplePDF documentation gleaned from files in the program directory. This is the topical content given the testers. The purpose of including this file is to illustrate the abundance material that can be reused.)

IMAGES
Snippets can be text or images
(From version 1.8 onwards, you can even place JPEG images in your publications. File extensions: .jpg, .jpeg, and .jif are recognized)

JPEG image files can be inserted into a Simple-PDF publication

you can use JPEG image files too (those files; obviously
cannot be edited using any text editor!)

TEXT
You can edit the individual snippets using a simple in-built text editor. You can also layout the snippets on individual pages. The editor is bare but it has the necessary features required of a text editor. We would love to develop a full fledged WYSIWYG tool for you -- but for that we need money

Remember that PDF would truncate any text that falls outside the snippet area. As the current version of Simple-PDF does not have text metrics capability, it may inadvertently crop your text.

Create multiple page PDF files from plain ASCII text files.

Why would you want to use external stream files, you may wonder? It can be very useful when you are creating several PDF files all of which share some common text files.

The editor also has a useful utility to insert PDF statements within your text. We've given the most popular ones. For annotations and links, you would have to construct special statements directly in your snippet stream as explained in simplepd.pdfs

If you want to shuffle pages around, you would have to edit the .SIM file separately in a text editor before using it in The Composer.

At the simplest level, a stream can be any ole ASCII text file. When you want more sophistication, you can selectively put native PDF marking operators into the text file. As you get into more and more sophistication, you would embed more and more PDF marking operators into your stream.

It may be advantageous to churn out one long text stream file at first. There is a chance that all of that would not fit into one page. Once you are ready to get into the publication design of your project, you could then divide that large stream file into parts and create separate stream files for each part; and use them separately on consecutive pages in your publication.

After you decide on the snippet size, location and page size of your PDF publication, create one stream text file for each of the snippet on one page. Then check out if the margins are not being exceeded in any direction. If the stream flows out of the margins, Acrobat reader would simply crop it out, so you must know how much to type on each line in your stream file.

It would help if you can use a font in your text editor that matches the one that would be finally used in your publication. For example; I use MS-SansSerif in my editor, to somewhat match up to Helvetica that is used by Acrobat Reader

You SHOULD NOT use automatic word-wrap feature in your text editor for your streams, if such a feature is available. Instead, you must create individual lines in the text file by consciously pressing the ENTER key at the end of each line. Simple-PDF does not have the ability to break lines as per the margins available. You would have to set the margins by trial and error.

If you have a very small snippet, then it may not be worth creating a separate text file for its stream. You can simply write the stream directly into the .SIM file instead of the stream filename.

Expected changes in future versions
Automatic wordwrapping and automatic formation of repeating snippets in newer pages, as per font used

SNIPPET

Simple-PDF uses the concept of snippets and pages to create any complicated PDF book.

Every PDF publication created by SIMPLE-PDF would consist of 3 kinds of elements: Pages, Snippets and Streams

Insert vector graphics inside snippets

Insert bitmap graphics (JPEG format) as snippets.

You can edit the individual snippets using a simple in-built text editor. You can also layout the snippets on individual pages

Each snippet appears on the page as a gray, empty dialog box with the Snippet number in the bottom left corner.

Double click on it to edit its contents. Resize the snippet by clicking and dragging the mouse at the corners of the snippet 'dialogs' - just like the way you would do it with other regular Windows. Remove unwanted snippets by clicking its close box.

When you want to add a new snippet (i.e. a rectangular area on the screen), make sure that the drop down list next to the 'Snippet' button is pointing to 'New'. If it was showing some other snippet number, then The Composer would attempt to use that same snippet on the currently active page

If it had already been used on that page, then such an action would only bring the earlier snippet forward of the others -- it will not allow you to use the same snippet on the same page twice. Using the file and folder controls at the top of the controls area of The Composer, you could also drag a filename onto the white region. When you leave the mouse, a snippet would be created for you at the location where you had lifted the mouse button. When you create a snippet in this fashion, The Composer would keep an indirect link to the contents (we use the term 'stream') of the snippet. The previously described method of clicking the 'snippet' button, would generate a snippet whose contents is an 'internal stream' which means, the contents of such snippets are not preserved separately in different files.

Sizing snippets

Each snippet is given a unique reference number -- which is seen at the bottom left corner of the snippet. The same number is displayed at the bottom of the text editor also. The size and location of each snippet is indicated in its title bar. The first two values indicate the bottom left corner (and NOT the top left corner) of the snippet. The other two values are the width and height in inches. You can easily resize the snippets by clicking and dragging on the corners or edges -- just like

any Windows 95/98 windows. You can delete a snippet by simply closing on the small 'x' at the top right of the snippet. In such a case, the snippet can be re-created by selecting that number from the 'Snippet' drop down list and clicking on the 'Snippet' button adjacent to it. The contents (i.e. stream) of such deleted snippets, are not lost if you now decide to bring the Snippet back to life after deleting. However, any saving action would flush out all the deleted snippets completely. If the stream was preserved as an external file; then that would be left untouched even if the snippet using it was deleted. At the bottom of the main screen, the salient details of the currently active snippet are also shown.

From left to right, they are: the rotation value, the stream contents (or the filename, if it is an external stream) and once again, the bottom left corner (X,Y) as well as the width and height of the snippet. Double-clicking on any of those sizing information can also change the size and location of the active snippet. If you change the rotation to any angle within, a colored strip would appear at the bottom of the snippet indicating that the indicated snippet would be finally rotated at the specified angle. Please note: Rotated snippets would not appear in the same rectangular region as shown within The Composer. The size and position of the snippet BEFORE rotation is what is shown.
Please also note that you would have to click inside a particular snippet at least once, for the program to register the new values. Shifting from one page to the other will NOT make the program automatically shift the snippet information it is showing: If you now double click at the 'Stream:' edit box at the bottom of the screen, you may still be accessing the earlier snippet and not the new one that is visible on the newly selected page. In short, when in doubt, click on the snippet once before editing

Editing snippets

To edit the contents of any snippet, whether internal or external; simply double-click on the snippet itself. The Composer meticulously preserves the last editing caret location, so even if you shift from one snippet stream to another, the editing caret will be faithfully waiting for you at the same location when you return. You can also double click on the 'Stream' part at the bottom of the screen to edit the contents of the snippet -- even if it was an external file. It is recommended that you resize the width of the editor to suit the line length you may want. You may have to produce some trial PDF files before you get comfortable. Remember that PDF would truncate any text that falls outside the snippet area. As the current version of Simple-PDF does not have text metrics capability, it may inadvertently crop your text.

Multiple pages from one snippet

From version 1.5, onwards, you can create multiple-pages from one snippet itself! Just write down: <page> where you need the page break to occur. At that point, the program would generate a new page; using the same snippet size parameters. (Note: There is NO space after the < and before the > characters, and such a command should be placed at the beginning of a new line, and there should NOT be any other characters on that line, even blank spaces! If you are wondering how I managed to write down the <page> above; this is how it happened: I added a few space characters just after the <page> Thus that particular line was NOT translated into a page-break.) Other snippets that you may have designed on that page would not get replicated when the multiple pages are formed. This feature can be very useful when you are writing a book with several chapters, and you do not know the number of pages in each chapter. Simply create a separate page for each chapter; and in the main snippet on the 'chapter-page' use the <page> feature to create multiple pages! You can see this feature used in the composer.sim file. Go to page 5 of the SIM file and examine the contents of the main snippet. You would notice that this particular feature enables the program to break it into two separate pages. Thus the generated PDF file would have one extra page than what was seen in the .SIM file When you use this feature; remember the following points:
a) Many times, we would start the page with some pdf(...) font commands. These would have to be repeated after each such page-breaks, else you would notice that the font reverts back to the default 10 point helvetica.

b) It is not advised to use the page break command for more than one snippet in any given designed page in the .SIM file (By the term 'designed page' I mean the page, as it appears in the .SIM file. When the .PDF file is eventually generated, it could have many more pages because of this 'page break' feature.
c) When a page break is discovered within a snippet; the program would break into a new page using the same snippet location and size. Other snippets are not used if their 'Reloc' parameter is 0 (zero). You can change this behaviour by using a 'Reloc' value of 1 for a snippet that needs to be repeated for every page automatically created using such page-breaks
d) You can insert page numbers for such generated pages by using the $ sign followed by the small case letter 'p'. Thus; this is page 2. (See the .SIM file to understand what I wrote here) If you notice; such a page number starts at 1 for every such snippet containing pagebreaks. Load novel.sim to see how you could use the page-break feature for writing a novel...

When you do any file saving, the program would flush out all unused snippets and all the snippets would get renumbered, if necessary.

A 'Snippet' is a rectangular part of a page. The same snippet can be used on many different pages, and if so required it can be relocated in different locations on those pages. However, a snippet can be placed only once on any individual page. You can have as many snippets as you need on a page.

You must have at least one snippet instruction in your .SIM file and you can have as many as you require.

The 'indexNumber' is a number used for indicating each snippet. You must use contiguous numbers for your snippets starting from 1. If you do not, you would get a corrupted PDF file. The index number of each snippet should also be unique.

The same snippet may be used in many pages (for example; as title blocks, etc.)

Multiple pages can be created from each snippet. Just insert <page> on a newline all by itself (no spaces before or after that <page>). Acrobat Reader is called automatically for previewing in Simple-PDF Composer

Repeating snippets
(Create any block and make it repeat on all your pages, without you having to redo any work. In fact the 'snippets' feature is central to the working of SIMPLE-PDF. The Vertical logo on the side of the pages is an example of a repeating snippet) Snippets can be text or images (From version 1.8 onwards, you can even place JPEG images in your publications. File extensions: .jpg, .jpeg, and .jif are recognized)

STREAM

Every PDF publication created by SIMPLE-PDF would consist of 3 kinds of elements: Pages, Snippets and Streams

But the third concept, a 'stream' talks about the actual contents that go into the PDF publication. Just like a 'stream' in the real world, it flows! But this time it brings in text and graphics into a snippet.

A stream, as stated before, is some data that flows into a snippet. At the simplest level, a stream can be any ole ASCII text file. When you want more sophistication, you can selectively put native PDF marking operators into the text file. As you get into more and more sophistication, you would embed more and more PDF marking operators into your stream.

The Composer would keep an indirect link to the contents (we use the term 'stream') of the snippet.

clicking the 'snippet' button, would generate a snippet whose contents is an 'internal stream' which means, the contents of such snippets are not preserved separately in different files. The data of 'internal streams' are therefore stored within the .SIM file.
Why would you want to use external stream files, you may wonder? It can be very useful when you are creating several PDF files all of which share some common text files.

You can also double click on the 'Stream' part at the bottom of the screen to edit the contents of the snippet -- even if it was an external file

For annotations and links, you would have to construct special statements directly
in your snippet stream as explained in simplepd.pdf

Annotations and links are to be created using the native commands in individual streams

Normally, when you start work on your publication, you should first be concentrating on the various streams that go into the snippets of your PDF publication. These stream files can be created in any ascii text editor that handles multiple files. From ver 1.3 onwards, Simple-PDF also contains its own interactive program, Simple-PDF Composer; that will do nicely.

Apart from the .SIM file, you would usually need a set of ascii text files for each of the streams used in your publication. The names of those stream files would be written down in the appropriate locations within the instructions found in the .SIM file

It may be advantageous to churn out one long text stream file at first. There is a chance that all of that would not fit into one page. Once you are ready to get into the publication design of your project, you could then divide that large stream file into parts and create separate stream files for each part; and use them separately on consecutive pages in your publication.

You SHOULD NOT use automatic word-wrap feature in your text editor for your streams, if such a feature is available. Instead, you must create individual lines in the text file by consciously pressing the ENTER key at the end of each line. Simple-PDF does not have the ability to break lines as per the margins available. You would have to set the margins by trial and error.

SIMPLE-PDF produces unoptimised PDF files. It does not encrypt the streams, nor does it compress them.

Reading of stream files have now been optimized. Now they are read only once; and therefore you save time! Annotations and links can now be given in files that are asked during runtime.

A nifty JavaScript utility that works in Netscape or IE (both 4 and above) is included for creating/editing .SIM files. Double click on makesim.html and the unique Javascript application will run.

SIM files

The command line utility is also included with Simple-PDF Composer, and as both the programs use the same .SIM file, you can switch from one to the other to create and manage PDF publications. Simple-PDF stores a publication using the .SIM file format. Before this version, the only way to create a .SIM file is to read the documentation for Simple-PDF, understand the sample .SIM file that was provided and create such files yourself. From this version onwards, The Composer can be used to create the .SIM file for you. Thus the Composer is a companion tool for the Simple-PDF command line utility.

If you need convenience, then use The Composer. But if you already have a .SIM file ready, which you want to reuse then use the command line utility. The Composer is quite intuitive to use: Here are the steps to create a publication:
a) Create a set of pages using the 'Add Page' button.
b) Assemble snippets on each page. Each snippet appears on the page as a gray, empty dialog box with the Snippet number in the bottom left corner.
c) Double click on it to edit its contents. Resize the snippet by clicking and dragging the mouse at the corners of the snippet 'dialogs' - just like the way you would do it with other regular Windows. Remove unwanted snippets by clicking its close box.
d) Once you are ready, click on 'Create PDF...' to generate the PDF file.
Before quitting, click on the 'Save SIM file...' button so that the publication can be saved as a SIM file, to be reused later.

The data of 'internal streams' are therefore stored within the .SIM file.

It is not advised to use the page break command for more than one snippet in any given designed page in the .SIM file (By the term 'designed page' I mean the page, as it appears in the .SIM file. When the .PDF file is eventually generated, it could have many more pages because of this 'page break' feature.

Before you quit the program, it is better that you save the .SIM file so that you can work on the same publication some other time. Having generated a PDF file will not give you the freedom to work on the publication again: You would require the .SIM file from which the PDF was generated If you need a tutorial, you could load composer.sim into The Composer. That is the SIM file that was used to create this PDF file. If you need to learn the internal details of the .SIM files, please read 'Simplepd.pdf' That document contains complete details on the .SIM file format. There are a few things that Simple-PDF Composer is currently not capable of directly handling, and these features would be incorporated in future versions. These include:
a) Using a GUI to handle Table of Contents (Bookmarks). Currently all advanced commands can be given within The Composer, but you would have use the native .SIM instructions
b) If you want to shuffle pages around, you would have to edit the .SIM file separately in a text editor before using it in The Composer.
c) Annotations and links are to be created using the native commands in individual streams
d) Simple-PDF is also capable of handling vector graphics. This requires knowledge of some parts of the PDF file format. We are in the process of creating Simple-PDF Publisher which will have all these features built in.

Apart from the .SIM file, you would usually need a set of ascii text files for each of the streams used in your publication. The names of those stream files would be written down in the appropriate locations within the instructions found in the .SIM file

The SIM instruction file
This file contains the actual instructions for forming the structure of the PDF publication. Each instruction is to be given in a separate line, exactly as indicated below. Even if the instruction becomes very long, each instruction MUST NOT SPAN more than one line.

You must have at least one snippet instruction in your .SIM file and you can have as many as you require.

Dimensions and sizes in Simple-PDF instructions (both in the .SIM file and the stream files) are given in inches (excepting for font size which is always in 'points'). But pure PDF marking operators use a units that scale this way: 72 units = 1 inch. You must appropriately calculate the actual values when you use pure PDF operators in the pdf(...) instruction embedded in your stream file. Also, the pdf(...) instruction should end in a space: For e.g. Notice a space after rg: pdf("0.3 0.1 0.8 rg ") If not, each pdf(...) would nudge into the next one.

In SIMPLE-PDF you would have to first print out the page containing the links, physically calculate the rectangles that you would need by measuring it off the printed page and then revise the stream files to incorporate the link rectangles. Actually it is no fault of SIMPLE-PDF, Adobe has designed PDF to be that way.

SIMPLE-PDF as presented in this fashion (i.e. using SIM files and stream files) could be argued as replacing one complex format (PDF) with another (SIM). To be productive with SIMPLE-PDF it would have been best to have a front-end that would do the physical positioning of the snippets, etc. (Simple-PDF Composer is such a front end)

a complete publication can be stored as one .SIM file. This was not possible in earlier versions, where large streams had to be stored as separate files. Simple-PDF now has a companion utility called Simple-PDF composer that can be used to create .SIM files interactively

A nifty JavaScript utility that works in Netscape or IE (both 4 and above) is included for creating/editing .SIM files. Double click on makesim.html and the unique Javascript application will run.

Both the command line utility as well as The Composer, uses the same .SIM file format for creating PDF files; and they can be used interchangeably

PAGE

Simple-PDF uses the concept of snippets and pages to create any complicated PDF book.

Create multiple page PDF files from plain ASCII text files.

Arrange rectangular regions called snippets on individual pages

Create a set of pages using the 'Add Page' button.

About pages

When you create a page, The Composer would show you a large white region reflecting the size of the page that was last set. If you need another size for the page, you can set the size using the controls just above the 'Add Page' button. Or you can change the page size by editing the width and height and double clicking on either of the edit boxes. Each page maintains its own page size separately. Clicking 'Del' to delete a page would prompt you for confirmation. If you had pressed the 'Shift' key, then a page would be deleted without any prompt. If you press 'Ctrl' when you click on Del, then all pages would get deleted, after asking you for a confirmation

Multiple pages from one snippet

From version 1.5, onwards, you can create multiple-pages from one snippet itself! Just write down: <page> where you need the page break to occur. At that point, the program would generate a new page; using the same snippet size parameters. (Note: There is NO space after the < and before the > characters, and such a command should be placed at the beginning of a new line, and there should NOT be any other characters on that line, even blank spaces! If you are wondering how I managed to write down the <page> above; this is how it happened: I added a few space characters just after the <page> Thus that particular line was NOT translated into a page-break.) Other snippets that you may have designed on that page would not get replicated when the multiple pages are formed. This feature can be very useful when you are writing a book with several chapters, and you do not know the number of pages in each chapter. Simply create a separate page for each chapter; and in the main snippet on the 'chapter-page' use the <page> feature to create multiple pages! You can see this feature used in the composer.sim file. Go to page 5 of the SIM file and examine the contents of the main snippet. You would notice that this

particular feature enables the program to break it into two separate pages. Thus the generated PDF file would have one extra page than what was seen in the .SIM file When you use this feature; remember the following points:
a) Many times, we would start the page with some pdf(...) font commands. These would have to be repeated after each such page-breaks, else you would notice that the font reverts back to the default 10 point helvetica.
b) It is not advised to use the page break command for more than one snippet in any given designed page in the .SIM file (By the term 'designed page' I mean the page, as it appears in the .SIM file. When the .PDF file is eventually generated, it could have many more pages because of this 'page break' feature.
c) When a page break is discovered within a snippet; the program would break into a new page using the same snippet location and size. Other snippets are not used if their 'Reloc' parameter is 0 (zero). You can change this behavior by using a 'Reloc' value of 1 for a snippet that needs to be repeated for every page automatically created using such page-breaks
d) You can insert page numbers for such generated pages by using the $ sign followed by the small case letter 'p'. Thus; this is page 2. (See the .SIM file to understand what I wrote here) If you notice; such a page number starts at 1 for every such snippet containing pagebreaks. Load novel.sim to see how you could use the page-break feature for writing a novel...

The editor also has a useful utility to insert PDF statements within your text. We've given the most popular ones. For annotations and links, you would have to construct special statements directly in your snippet stream as explained in simplepd.pdf

If you want to shuffle pages around, you would have to edit the .SIM file separately in a text editor before using it in The Composer.

Multiple pages
From ver 1.5 onwards, you can create multiple pages. Simply write <page> on a new-line all by itself wherever you need a page break. If you use $ p (the letter p shall immediately follow the $, without any space) then that is substituted by the page number created for multiple pages. Refer novel.sim for an example

It may be advantageous to churn out one long text stream file at first. There is a chance that all of that would not fit into one page. Once you are ready to get into the publication design of your project, you could then divide that large stream file into parts and create separate stream files for each part; and use them separately on consecutive pages in your publication.

The PDF file format is quite vast and covers all aspects of page creation. However the PDF file format is daunting for common tasks and it is almost next to impossible to convert simple text files (such as the one you are now reading) into a PDF publication simply by hand.

EDITOR

The editor is bare but it has the necesary features required of a text editor. We would love to develop a full fledged WYSIWYG tool for you -- but for that we need money.

It is recommended that you resize the width of the editor to suit the line length you may want.

The editor also has a useful utility to insert PDF statements within your text. We've given the most popular ones. For annotations and links, you would have to construct special statements directly in your snippet stream as explained in simplepd.pdf

If you want to shuffle pages around, you would have to edit the .SIM file separately in a text editor before using it in The Composer.

stream files can be created in any ascii text editor that handles multiple files. From ver 1.3 onwards, Simple-PDF also contains its own interactive program, Simple-PDF Composer; that will do nicely

It would help if you can use a font in your text editor that matches the one that would be finally used in your publication. For example; I use MS-SansSerif in my editor, to somewhat match upto Helvetica that is used by Acrobat Reader.

You SHOULD NOT use automatic word-wrap feature in your text editor for your streams, if such a feature is available. Instead, you must create individual lines in the text file by consciously pressing the ENTER key at the end of each line. Simple-PDF does not have the ability to break lines as per the margins available. You would have to set the margins by trial and error.

You can edit the individual snippets using a simple in-built text editor. You can also layout the snippets on individual pages. The editor is bare but it has the necessary features required of a text editor. We would love to develop a full fledged WYSIWYG tool for you -- but for that we need money.