Cl-Pdf And Cl-Typesetting Users Documentation

Written by: Brian Sorg, Founder Liberating Insight LLC
brian.sorg@liberatinginsight.com

Introduction

Cl-Pdf and Cl-Typesetting are cross-platform stand-alone Common Lisp Libraries for producing PDF documents. The libraries are overseen by Marc Battyani of Fractal Concept and can be downloaded from their website

http://www.fractalconcept.com.

Cl-Pdf and Cl-Typesetting are both released with a FreeBSD style license making them usable for commercial work.

For the purpose of this documentation it will be assumed that Cl-Pdf and Cl-Typesetting are really one module used in conjunction to produce PDF documents. In reality Cl-Pdf is the base library for producing PDF documents and has capabilities similiar to that of a CAD system, ie manually positioning text boxes on the page and then filling them with content. As a follow on Cl-Typesetting builds on Cl-Pdf offering a complete typesetting system. Therefore most users will focus on implementing the functionality of CL-Typesetting and will only utilize Cl-Pdf directly when producing some specific document features.

This document is intended it help programmers utilize Cl-Pdf and Cl-Typesetting to produce pdf documents from their Lisp applications. It is not a developer's guide to the source code for Cl-Pdf and Cl-Typesetting. If you have a specific question about the code or would like to contribute please use the cl-pdf-dev mailing lists to post your contributions and questions.

Also note that the documentation is not intended to function as a stand alone document. Instead being entirely produced utilizing Cl-Pdf and Cl-Typesetting, it is designed to be read in conjuction with the source code that produced it. Therefore you will find very few code examples in the text. If you wish to see how something was generated, take a look at the source code that produced the document. It is logically broken up starting with the main function being located in the document.lisp file, each chapter being contained in a file named chapter-*.lisp and document wide functions and macros being found in the infrastructure.lisp file

Obtaining

Cl-Pdf and Cl-Typesetting can be downloaded from the Fractal Concept company website at:

http://www.fractalconcept.com.

Tarball versions of the lastest stable releases are available, but it is recommended that you download the latest code from the Subversion repository. Directions of how to do this can be found on the website

Installation

To install Cl-Pdf and Cl-Typesetting, it is recommended that you use the asdf system definitions provided with the Cl-Pdf and the Cl-Typesetting packages. If you are not familiar with the asdf packaging system you can download it and its documentation from

http://www.cliki.net/asdf

Once you have loaded the asdf package. You can compile and load CL-Pdf and Cl-Typesetting with the following commands:

```
(push #P"<your path>/cl-pdf/" asdf:*central-registry*)
(asdf:operate 'asdf:load-op 'cl-pdf)
(push #P"<your path>/cl-typesetting/" asdf:*central-registry*)
(asdf:operate 'asdf:load-op 'cl-typesetting)
```

As part of loading the system there are a few important parameters that you will want to specify that will effect the default behavior of documents produced. The first ones are found in the Cl-Pdf package and are defined in the config.lisp file. There are many parameters found in the file, but the most important ones are:

max-number-of-pages	The maximum number of pages a document may contain before it stops. Default is 1000
default-page-bounds	What dimensions a default page in Cl-Pdf should have. The default defines an A4 size page with portrait orientation, options include *a4-portrait-page-bounds*, *letter-portrait-page-bounds*, *a4-landscape-page-bounds*, *letter-landscape-page-bounds*.

The other parameters you should become familiar with are found in the Cl-Typesetting package and are located in the top-level and specials files. Once again there are a number of parameters which you can use to customize your pdf documents. However, make sure that you have properly configured the following ones:

default-page-size	Controls the default page dimensions for Cl-Typesetting capabilities. Options include: :A3, :A4, :A5, :Letter, :Legal. The default is :A4
default-page-orientation	Controls the default page orientation. Options are :portrait and :landscape. The default is :portrait
default-font	Controls the default font type, The default is Helvetica
default-font-size	Controls the default font size, The default is 12

Ch 2: Getting Started

The following sections will define the basic process for creating a basic pdf text document.

The top level macro you will use to define a pdf document is the **tt:with-document** macro. This macro prepares everything for you that will be required to generate the document. So let us create our first pdf document. The following code snippet will create a pdf document containing one paragraph and write that document to a file with the path /tmp/first -doc.pdf. The source code to the example can be found in the cl-pdf-doc directory under examples/first-doc.lisp

```
(defun first-doc (&key (file "/tmp/first-doc.pdf"))
 (tt:with-document
     ()
   (let ((content (tt:compile-text
      ()
    (tt:paragraph () "Generated with Cl-Pdf and Cl-typesetting"))))
     (tt:draw-pages content)
     (when pdf:*page* (typeset:finalize-page pdf:*page*))
     (tt:write-document file))))
```

First Document: Including Text

So lets explain the details of what is occuring with this example. First, we see the use of the **tt:with-document** macro. The macro instantiates the objects and initializes the parameters required to produce the document. The second step is to prepare the content of the document. This is achieved using the **tt:compile-text** macro. The compile-text macro sets the style parameters such as font, font-size, font-color, and background-color to the global settings for all of the content that is contained within the macro. It, in a sense, prepares the following text for inclusion in the pdf document. The next step is to take the prepared content and create pages in the document with it. This is accomplished with the **tt:draw-pages** function. It initializes the settings for the size, orientation and margins of the page; it will include any headers or footers that were provided; and will calculate as needed the page breaks required to include the content on the page. The next line ensures that the final page in the pdf document has been properly marked as finished and will not remain open waiting for more material. The last statement **tt:write-document** takes the prepared pages and writes them to the pdf document. The end result of this function then is a pdf file with one paragraph at the top of the page that says *Generated with Cl-pdf and Cl-typesetting*

Now lets extent this example by make some changes to the font settings. Before we do this though, it is necessary to understand some of the basics of pdf fonts. First, since pdf documents are supported acrossed a broad range of operating systems the document format could not rely on the font libraries provided by the various operating systems. To overcome this problem they created a small set of standard fonts that every pdf reader on every operating system is required to have, and they created a mechanism for attaching a font type to a specific document is specialized fonts were requires. Once a font is attached to a document all readers will have the information they require to display the text in this font to the reader. Cl-pdf supports both the standard and embedded fonts. Here hower we will only take a look at the standard fonts.

The standard fonts for pdf documents can all be found in the Cl-pdf code base under the directory called afm. In this directory you will find the font definition files. In practical terms there are only three font types, Times-Roman, Helvetica, and Courier and four font styles for each type, standard, bold, italic, and bold and italic.

Using the tt:paragraph function you can change the font simply by providing the keywords *:font* and *:font-size.* For example by adding the following lines to the first-doc function after the initial paragraph statement, you can experiment with 3 new font types and sizes

```
(tt:paragraph (:font "Courier" :font-size 16) "Courier Font size 16")
```

```
(tt:paragraph (:font "Times-Bold" :font-size 8) "Times in bold with font size 8")
(tt:paragraph (:font "Helvetica-Oblique" :font-size 14) "Helvetica italic font size 14.5")
```

Example 2: Font Types and Sizes

As you can see from the example, the font types are defined by using the file name found in the afm directory without the .afm extension. The font-size is defined in pts, where 72 pts is equivalent to 1 inch. Decimals are permitted.

Finally, lets look at generating some output to include in the document. In general, dynamic content may be added simply by generating a string within the context of a typesetting document formating structure. To demonstrate this, we will add numbered paragraphs to the first-document example started above. Simply add the following code snippet after the last paragraph declaration but within the compile-text macro.

```
(dotimes (i 10)
  (tt:paragraph ()
    "Paragraph # " (tt:format-string "~d" i)))
```

First Document: Generating Content

In this example ten paragraphs are created, each followed by the number of the paragraph.

That covers all of the basics required to get started with Cl-Typesetting. From here on out it is just a matter of learning the different macros and functions available to help arrange and format the contents of your documents. These explanations following (or will follow depending on when you are reading this) in Chapters 3 and 4.

More Information

As one progresses through this documentation it will also be helpful to review examples from other sources. Both Cl-pdf and Cl-typesetting include a number of examples for one to try out and experiment with. They can be found in the source directories. For Cl-Pdf they are in the examples/examples.lisp file. In Cl-typesetting they are in the test.lisp file. You may also want to review the source code used to generate this documentation. It is found under the doc/source in the cl-pdf directory. Also the source for the examples used through out the documentation are located in under examples in the doc direction.

If you are interested in understanding how a PDF document is constructed, you can review the file specification yourself. At the time of this writing Adobe published it under http://www.adobe.com/devnet/pdf/pdf_reference.html.

Ch 3: Document & Layout Functions

TT:WITH-DOCUMENT

Syntax

macro **tt:with-document** *(&rest args) &body body* ==> nil

Arguments

&rest
- *:author*: Author of the document.
- *:title*: Documents Title
- *:keywords*: Keywords of the document
- *:subject*: Subject of the document

&body
- *body*: Code generating the document

Description

The *with-document* macro is the starting point for all documents. It is responsible for initializing all of the required objects need to generate the document. The arguments for author, title, keywords, subject are filled in a meta data for the document, and will appear under the document properties listings.

TT:WRITE-DOCUMENT

Syntax

defmethod **tt:write-document** *output-location &optional document* ==> nil

Arguments

Required
- *output-location*: string, pathname or stream, determines where the document content should be sent

&Optional
- *document*: default *document, normal users will not need to provide this value

Description

Used to write out the given content of the document

TT:DRAW-PAGES

Syntax

function **tt:draw-pages** *content &rest args &key (size *default-page-size*) (orientation *default-page-orientation*) bounds margins header (header-top *default-page-header-footer-margin*) footer (footer-bottom *default-page-header-footer-margin*) break finalize-fn &allow-other-keys)* ==> nil

Arguments

Required
- *content*: A section of the doucment prepared by the macro compile-text

&key
- *size*: Dimensions of the document's page
- *orientation*: Orientation of the document
- *bounds*: Media box; overwrites size and orientation when specified.
- *margins*: White space between the page's edge and the text, list of 4 numbers defining left-margin top-margin right-margin bottom-margin

header: Content or function of ftype (function (page) content which defines the header content of each page
header-top: Distance between the top media edge and the header.
footer: Content or function of ftype (function (page) content which defines the footer content of each page
footer-bottom: Distance between the bottom media edge and the footer.
break: Force new page ::= :before | :after | :always (both ends)

Description

Top level function useful for generating a multi-page section of the document. This function will start generating the content and will automatically overflow content onto another page as required to include all of the content in the document.

TT:PARAGRAPH

Syntax

macro **tt:paragraph** *(&rest style) &body body* ==> nil

Arguments

&rest

style: These are all of the text style markers, used in the paragraph. The common markers are defined as follows
:top-margin: space between the text of paragraph and the preceding document content
:bottom-margin: space between the content of the paragraph and the following document content
:first-line-indent: Number, the indentation of the first line of the paragraph, default is 0
:font: designator of which font to utilize in this paragraph
:font-size: size of the font to employ
text-x-scale:
:color: Foreground/text color
:background-color:
h-align: Content alignment
left-margin:
right-margin:
pre-decoration:
post-decoration:

Description

Defines a paragraph with the given style set for the contents of the paragraph.

Appendix A

Parameters

default-font	type	Controls the default font type, The default is Helvetica
default-font-size	type	Controls the default font size, The default is 12
default-page-bounds	pdf	What dimensions a default page in Cl-Pdf should have. The default defines an A4 size page with portrait orientation, options include *a4-portrait-page-bounds*, *letter-portrait-page-bounds*, *a4-landscape-page-bounds*, *letter-landscape-page-bounds*.
default-page-orientation	type	Controls the default page orientation. Options are :portrait and :landscape. The default is :portrait
default-page-size	type	Controls the default page dimensions for Cl-Typesetting capabilities. Options include: :A3, :A4, :A5, :Letter, :Legal. The default is :A4
max-number-of-pages	pdf	The maximum number of pages a document may contain before it stops. Default is 1000