Data Driven Document Generator User Instructions

dddg was written to generate membership cards for a group of people, with the input data (names etc) read from a csv file. I realised as writing it that it could be more generic, and have made the card layout data-driven from an xml file. You are welcome to use it for any purpose, but please see the small print regarding the third-party library licences.

Installation

The program java source code and example configuration files are available on github:

https://github.com/LouGifethBSc/DataDrivenDocumentGenerator

To run the program, the source code must be downloaded and built into a runnable jar file.

The following third party libraries are required:

- Apache Log4j2, https://logging.apache.org/log4j/2.x/
- Apache PDFBox, https://pdfbox.apache.org/download.cgi
- W3C DOM XML Parser, (part of standard JRE?)

Runtime Dependencies

Java runtime environment (JRE). Please check that you have the latest version.

Running

To run the program from a command-line prompt, type:

java –jar dddg.jar

All inputs will be read from the configuration file **config.txt**

Or you can specify your own config file:

java –jar dddg.jar your-config-file

The config file contains the parameters in the format **name=value**

name	**description**	**default value**
input_csv	The csv (comma separated variable) file containing the data to be processed – see below for format	input.csv
max_rows	The maximum number of rows (excluding the header row) to be processed from the input csv file. Set this to 1 to check the config and formatting	300
id_column_name	The title of the column in the csv file containing the id field. This column is used by the programme to give each card a unique name	id
card_layout	Definition of the card layout – see below for	card_layout.xml

	details and examples	
card_name_prefix	The cards will be named using this prefix and the individual id from the csv file, with the extension .pdf	card_
font_dir	The folder where truetype fonts are located	C:\Windows\Fonts

Input csv file format

The input csv file must have one header row, and one row of data per card.

The format has been kept as flexible as possible. The only column that is required is **id**. The config file is used to specify the column title for this item – see above. The columns can be in any order. Titles of all other columns must be present, but can be anything you like.

The data in the **id** column is used in naming the cards, so should contain unique values, otherwise cards will be overwritten.

A csv file is a text file containing comma separated variables. A spreadsheet may be converted to a csv file easily by saving in csv format.

For example:

ID,First name,Surname,Email Address,Date subs next due,ice_name,ice_phone
0001,The ,Chairman,chair_email@somewhere.com,30-06-18
0042,Lou,Gifeth,lou.gifeth@somewhere.com,30-06-18,Thomas Wolsey,01473 210055

In this example, the id column title would be specified in the config file as:

id_column_name=ID

Card Layout

The card layout is completely flexible, being specified by the card layout file.

The card layout file is defined in xml format, this being easily read by both people and computers. For example, the text “Hello World” could be defined as:

```
<text>
        <position>108,130</position>
        <text>Hello World</text>
</text>
```

Where the position is defined as the x,y co-ordinates relative to the bottom-left corner of the card. All sizes are defined in points (as in font sizes), there are 72 points in one inch.

The text font size is 10 by default, but can be specified in the layout file in one of two ways.

To define the font size for all subsequent text elements:

```
<font>
        <size>12</size>
</font>
```

To define the font size for the current text element:

```
<text>
        <size>12</size>
        <position>108,130</position>
        <text>Hello World</text>
</text>
```

The following elements can be used within the card layout file:

element	description	example
card_size	Defines the card size (x,y)	<card_size> <size>243,153</size> <margin>9</margin> </card_size>
font	Defines the font for text elements	 Helvetica <style>Bold</style> <size>12</size>
rectangle	Draws a rectangle	<rectangle> <position>0,0</position> <size>243,153</size> </rectangle>
image	Includes an image from a file	<image> <file>ioglogo.png</file> <position>9,54</position> <size>90,90</size> </image>
text	Adds text	<text> <position>108,130</position> <text>Hello World</text> </text>

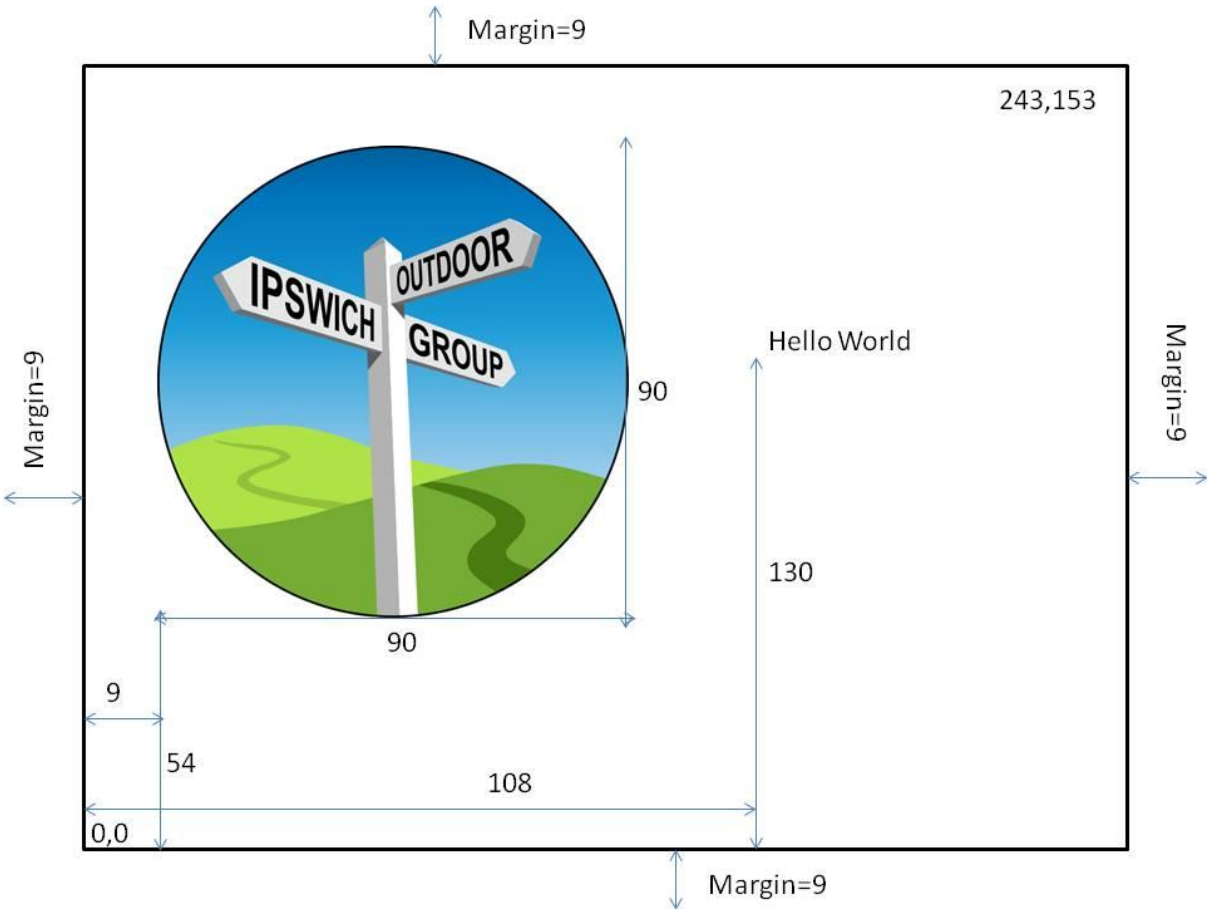
Sizes, margins and positions

All sizes are defined using points, there are 72 points in one inch. This is the standard way of defining font sizes.

The above example defines a card the same size as a standard credit card, 3.375x2.125 inches, which equates to 243x153 points.

The optional <margin> tag is used to define a blank margin around the card, which could be useful when printing and cutting out.

Positions are measured from the bottom left hand corner of the card (ignoring any margin). Layout example using the examples in the table above:



Fonts

The program supports the 3 standard PDF fonts (**Times**, **Helvetica**, **Courier**), also known as type 1 fonts. The **<style>** tag can be used to select one of: **Bold**, **Italic**, or **BoldItalic**.

The program also supports truetype fonts (type 0), which must be located on the host computer running the program. By default, the program will look in the **C:\Windows\Fonts** folder, but this can be overridden by setting **font_dir** in the config file.
For truetype fonts, the style is included as part of the font filename, e.g. **arial**, **arialbd** (arial bold), **ariali** (arial italic), **arialbi** (arial bold italic).

Personalising the card

From the examples above, all cards will be the same. Cards may be personalised by referring to the data in the input csv file within the card layout. For example, to include the ID field which is in the column with the title “ID”:

```
<text>
        <position>108,99</position>
        <value>ID</value>
</text>
```

Text elements may be combined by listing the fixed text and the values to be read from the csv file in order. For example, to include a name field made up of two columns titled “First name” and “Surname”:

```
<text>
        <position>108,130</position>
        <value>First name</value>
        <text> </text>
        <value>Surname</value>
</text>
```

Note the space between <text> and </text> which will separate the first name and surname.

Long text that won’t fit

Text can be split over two lines using the <max_length> and <overflow> tags. This could be useful for people with very long names. For example:

```
<text>
        <position>108,130</position>
        <text>Longfirstname Evenlongerdouble-barrelledsurname</text>
        <max_length>25</max_length>
        <overflow>108,117</overflow>
</text>
```

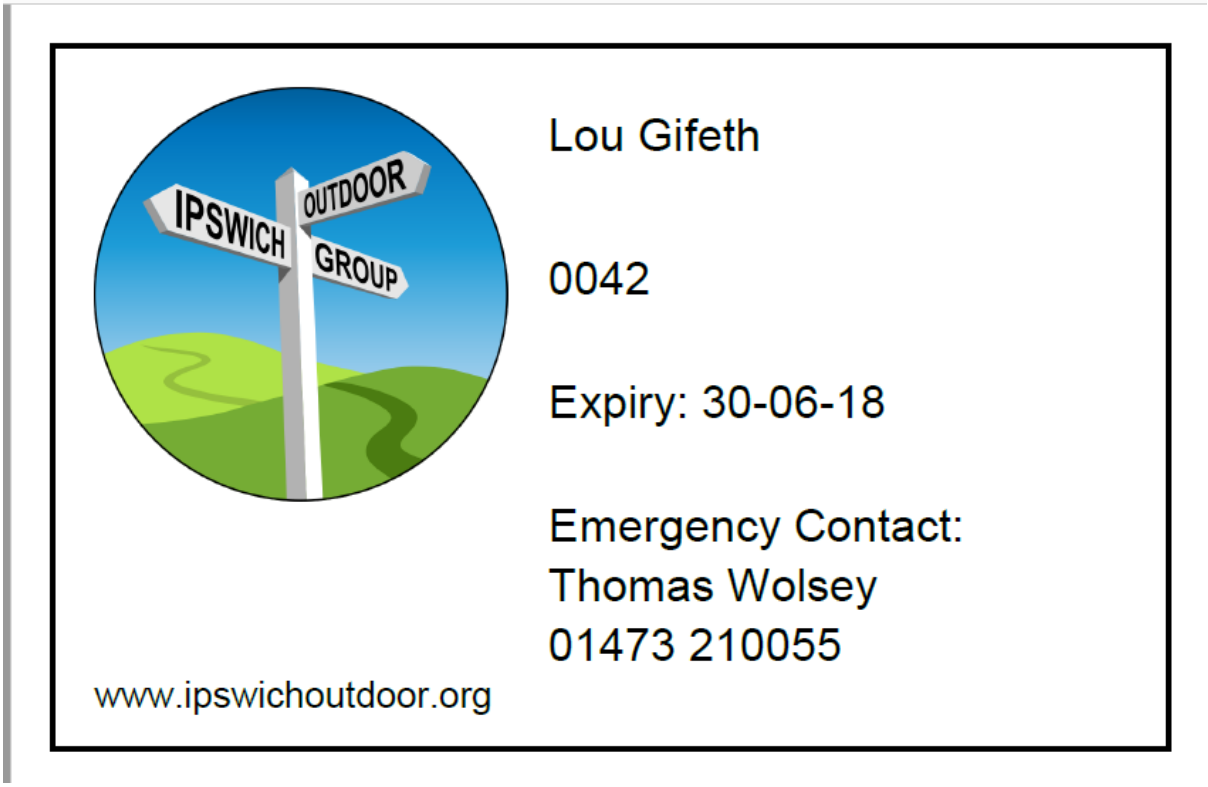
Bear in mind when specifying <max_length> that most fonts are not fixed width, i.e. different letters take up different amounts of space.

The program will try to split the text at a space (this would typically put first name and surname on separate lines). Note that in the above example, “**Evenlongerdouble-barrelledsurname**” is still longer than the max length of 25. That’s just too bad, it will be allowed to overflow.

Full layout example

```
<card>
<card_size>
        <size>243,153</size>
        <margin>9</margin>
</card_size>
<font>
        <font>Arial</font>
        <size>10</size>
</font>
<rectangle>
        <position>0,0</position>
        <size>243,153</size>
```

```
</rectangle>
<image>
        <file>ioglogo.png</file>
        <position>9,54</position>
        <size>90,90</size>
</image>
<text>
        <position>108,130</position>
        <value>First name</value>
        <text> </text>
        <value>Surname</value>
        <max_length>25</max_length>
        <overflow>108,117</overflow></text>
<text>
        <position>108,99</position>
        <value>ID</value>
</text>
<text>
        <position>108,72</position>
        <text>Expiry: </text>
        <value>Date subs next due</value>
</text>
<text>
        <position>108,45</position>
        <text>Emergency Contact:</text>
</text>
<text>
        <position>108,32</position>
        <value>ice_name</value>
</text>
<text>
        <position>108,19</position>
        <value>ice_phone</value>
</text>
<text>
        <position>9,9</position>
        <size>8</size>
        <text>www.ipswichoutdoor.org</text>
</text>
</card>
```



Error Messages

I have tried to make the error messages understandable where possible, but no doubt there are lots of things that could go wrong that I haven't thought of or checked for.

The program uses the standard log4j2 logging framework, so you could try setting parameters in the **log4j2.xml** file to get more info. (Hint: change "error" to "trace")

Here are some example error messages I have come across while testing. If you come across any others please let me know and I will add to the list.

Most errors will be followed by more detailed info (a stack trace) which may be helpful (or not).

Error message	**Cause/solution**
Error: Could not find or load main class xxx	Mistyped xxx or left off the –jar flag **java –jar dddg.jar**
Error: Unable to access jarfile dddg	Missed off the .jar **java –jar dddg.jar**
ERROR StatusLogger File not found in filesystem or classpath: log4j2.xml ERROR StatusLogger Reconfiguration failed: No configuration found for …	Log4j2 config file missing. By default, only errors will be logged. The program will continue running.
ERROR ...NameValuePairList – error reading parameters from file xxx	Config file missing or not readable. The program will continue running using default values.
ERROR ...dddg – error processing file xxx	Input csv file missing or not readable
java.io.FileNotFoundException:	Card layout file missing or not readabale

…/card_layout.xml (The system cannot find the file specified)	
[Fatal Error] card_layout.xml:15:3: The element type "xxx" must be terminated by the matching end-tag "</xxx>".	Syntax error in card layout file, the xml tag names must match e.g. <text>…</text> In this example error message, the error is at line 15 column 3.
javax.imageio.IIOException: Can't read input file!	Image file defined in card layout missing or not readable. The program will continue running but the image will be missing from the cards.
java.io.FileNotFoundException: xxxxxx.pdf (The process cannot access the file because it is being used by another process)	You probably have the file xxxxxx.pdf open in another program e.g. viewing a previously generated card in a pdf reader
`ERROR ...dddg.PDFFontHelper - file not found: C:\Windows\Fonts/xxx.ttf`	The font file xxx.ttf is not found. The program will continue using the previously specified font.

Known problems

problem	**comment**
Some of the error messages are a bit cryptic e.g. if errors are made in the syntax of the card layout file	Sorry, I'm only doing this for a bit of fun.
Long text can overflow the boundary of the card, e.g. if someone has a very long name	The text can be specified to wrap to an overflow area . This will typically allow first name and surname to be on separate lines – see example. This should cater for all the names I've seen in the IOG membership list, but there is no guarantee.
The log4j configuration file is hard-coded	It has to be called **log4j2.xml**
Not tested yet with a large number of cards	May be problems with memory leaks or it may just take a long time. With a small number of cards, it takes less than a second to create each card. 200 cards at 1 second each would take 3 minutes 20 seconds.
Only tested on Windows computer (Windows 10)	The program should work on any computer that supports java, including Mac and Linux, but not tested.

The small print

This program is provided “as-is” and with no warranty or guarantees.

The author accepts no liability for any adverse effect through usage.

The program is provided free of charge, although the author would be pleased to accept beer if it proves to be useful.

Some third-party libraries have their own licences. It is understood that these are all ok for the intended usage.

Apache PDFBox and Log4j2 - https://www.apache.org/licenses/

Oracle JavaMail - https://javaee.github.io/javamail/LICENSE

W3C DOM XML Parser - http://www.w3.org/Consortium/Legal/2015/copyright-software-and-document

Change Log

Version	Date	Comment
0.1	24-08-17	For testing by IOG committee
0.2	18-09-17	Removed email capability and published on github
0.3	22-09-17	Added support for standard (type 1) and truetype (type 0) fonts

Contact details: lou.gifeth@btinternet.com