DocGen - a documentation tool

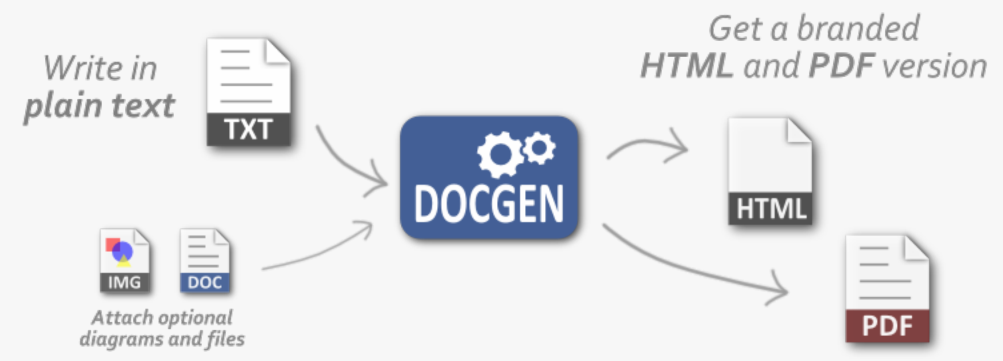
Owner: Mark Macdonald	**Version:** 2.0.0	**Released:** XX/XX/XXXX
Author: Mark Macdonald	**Contributors:**	
Module:	**ID:**	**Link:** Github

Summary

DocGen is a documentation tool for software products. It takes plain text or Markdown as input, and generates both a static website and a PDF copy.

Table of Contents

Better documentation for software products.


Write in plain text
TXT
IMG
DOC
Attach optional diagrams and files
DOCGEN
Get a branded HTML and PDF version
HTML
PDF

Download   Report Issues

DocGen is a **static website generator** that's ideal for making technical user guides for software products.

Self-contained website

Creates a static website that works on any server, or as local files (CD, shared drive etc).

Optional PDF

Also publishes the website content as a single PDF, using wkhtmltopdf.

Human-friendly input

Write content in plain text, or the human-friendly CommonMark format (Markdown).

Easy to version control

Easily version control software documentation in sync with its product.

Table of contents

Automatically creates document and page tables of contents, with links and PDF page numbers.

Code syntax highlighting

Automatically highlights code blocks, using Highlight.js, with language detection.

Mathematical expressions

Displays mathematical expressions natively in all browsers, using MathJax.

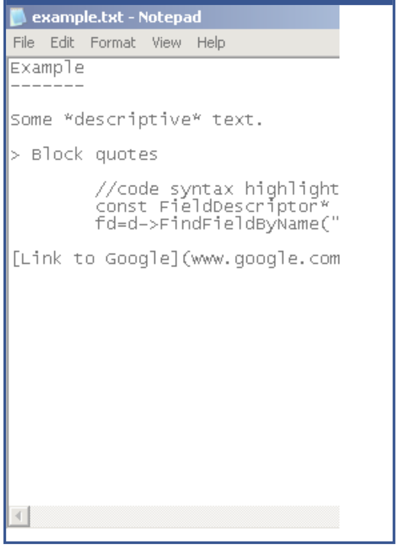
Branding and metadata

Easily brand with a logo, attribute ownership, and attach release notes.

DocGen makes it easy for anyone to create high-quality technical documentation.

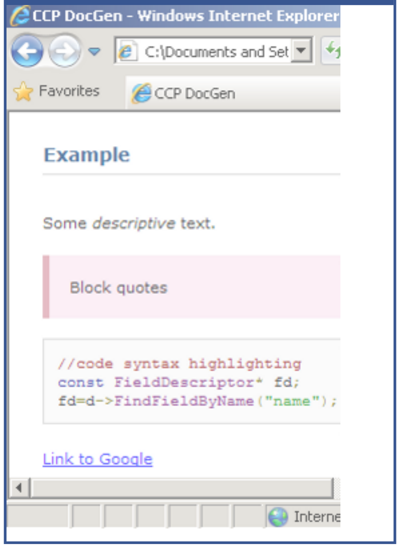
How it works

"You write in plain text or human-friendly human-friendly "



```
Example
-------

Some *descriptive* text.

> Block quotes

        //code syntax highlight
        const FieldDescriptor*
        fd=d->FindFieldByName("

[Link to Google](www.google.com
```

"DocGen styles and publishes all your content as a website"



Example

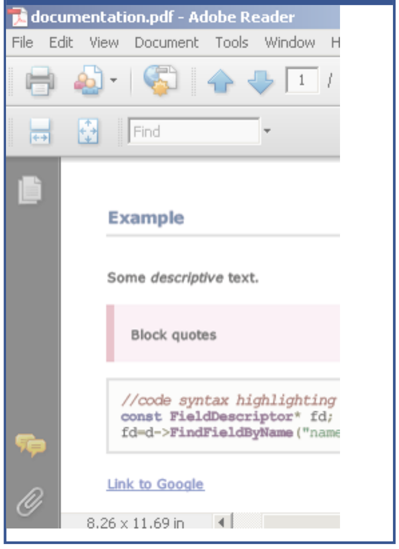
Some *descriptive* text.

> Block quotes

```
//code syntax highlighting
const FieldDescriptor* fd;
fd=d->FindFieldByName("name");
```

Link to Google

"DocGen also creates an equivalent PDF copy"


documentation.pdf - Adobe Reader
File Edit View Document Tools Window H
1
Find
8.26 x 11.69 in

Example

Some *descriptive* text.

> Block quotes

```
//code syntax highlighting
const FieldDescriptor* fd;
fd=d->FindFieldByName("name
```

Link to Google

Flexible Input Formats

- Plain text
- Commonmark (Markdown)
- HTML
- MathML
- Image diagrams
- Attach other documents

Configurable Metadata

- Branding (logo, title, organization)
- License, copyright, and legal markings
- Ownership and attribution
- Version information
- Release notes (change log)

> DocGen is probably not the right tool for precision PDF layout. Or for product configurators which need to output variants based on a common template. Docgen is intended more for free-form, human generated content which is regularly updated and refactored, and standardised for each product release.

Supports modern browsers

Output created by DocGen works on modern browsers (desktop and mobile devices).

- tested in Internet Explorer 9+, Chrome, Firefox, Safari

Quick Start

In just three commands:

- Install DocGen
- Scaffold an empty template
- Generate a static website

Enter these commands in the terminal:

```
npm install -g docgen-tool
node docgen.js --scaffold $HOME/example
node docgen.js -i $HOME/example -o $HOME/site
```

See the installation guide for more detailed instructions.

Installation

This section explains how to install DocGen.

Supported platform

'Supported platforms' means the software required to **run** the DocGen tool. The static website **produced** by DocGen will work on all modern browsers on all all major operating systems (see browser support).

The supported platform for this version of DocGen is: **Ubuntu 14.04** with **Node.js 0.12.0** and **wkhtmltopdf 0.12.2.1** (with patched qt).

While other operating systems and dependency versions may work, they are not tested or officially supported.

Dependencies

DocGen requires **Node.js**. To install on Ubuntu Linux, enter these terminal commands:

```
curl -sL https://deb.nodesource.com/setup_0.12 | sudo bash -
sudo apt-get install -y nodejs
```

This method uses the NodeSource Linux Repositories (recommended).

For other platforms (not supported), see the download page or vendor instructions.

Optional dependencies (only for PDF)

For optional PDF support, DocGen requires **wkhtmltopdf**. See the wkthmltopdf website for installation instructions.

Quick install with NPM

The quickest way to install DocGen is with **npm** (the JavaScript package manager). Enter these terminal commands:

```
npm install -g docgen-tool
```

Install by direct download

DocGen can also be installed by direct download.

Upgrading

This section explains how to upgrade from old versions of DocGen.

DocGen 1.0.* to DocGen 2.0.0

DocGen 2 is not backwards compatible with DocGen 1, but the upgrade is easy with these instructions:

- Install DocGen 2 by following the installation guide
- Create a replacement 'source' directory, e.g: `node docgen.js --scaffold $HOME/source`
- Migrate the source files
  - Copy the source (.txt) files from the old 'Src' directory to the new 'source' directory
  - Remove the top-level page headings in each source file (DocGen now inserts these automatically, based on contents.json)
  - Rename *'change-log.txt'* to *'release-notes.txt'*
- Migrate the attached content
  - Place a logo with filename 'logo.png' in 'files/images' directory (the logo is now part of the source)
  - Copy the images from *'Images'* to *'files/images'*
  - Copy attached files from *'Files'* to *'files'*
- Migrate the metadata
  - Edit *'parameters.json'* with the relevant values from *'doc-parameters.yml'*
  - Edit *'contents.json'* with the relevant values from *'table-of-contents.yml'*
- Run DocGen: `node docgen.js -i $HOME/source -o $HOME/output`
- Check the styling in the latest version still works well with the old content
- Note that the PDF is no longer generated by default (pass the *'-p'* option)

> DocGen 1 tool-behaviours.yml has been replaced by command-line options See `node docgen.js -h` for help.

Writing content

Overview

Using DocGen involves working with three folders:

- **Source** (the document master, where all your content goes)
- **Generator** (the only task you perform here is to run **Generator.rb** when publishing the content)
- **Output** (where the generated HTML website and PDF will appear)

'Source' and 'Generator' are included in the DocGen download, and 'Output' will appear the first time you run Generator.rb.

Writing content

DocGen takes any text file (extension .txt) in the Source folder and converts it. Each text file will be a separate page in the website and a separate chapter in the PDF. You can create as many new files as you wish (start by editing the example file called index.txt).

> Make sure you save content files (.txt) with **UTF-8 encoding**. This makes non-standard characters (ø © é etc) work.

You mostly write paragraphs of plain text using any text editor. Stylistic effects can be achieved by using a plain-text publishing format called Markdown. The best way to learn is to see an example (below), and then read the Markdown reference.

What you type:

```
Headings give structure
-----------------------

Paragraphs are big blocks of text with lots of sentences.
Some sentences have *significant* concepts or **very important ones**.

A new paragraph starts when the line before looks blank.
Some paragraphs will contain [example links].

    //To make a code block, just indent with a tab. "Hello World" in Ruby:
    5.times { puts "Hello!" }
```

What it looks like in the website and the PDF:

Headings give structure

Paragraphs are big blocks of text with lots of sentences. Some sentences have *significant* concepts or **very important ones**.

A new paragraph starts when the line before looks blank. Some paragraphs will contain example links.

```
#To make a code block, just indent with a tab. "Hello World" in Ruby:
5.times { puts "Hello!" }
```

Adding diagrams and other files

- Any diagrams (in image form, e.g. JPEG, PNG, GIF etc) should be put the Source/Images folder.
- Any other files you want to attach should go into Source/Docs.

See the Markdown reference for how to attach them. Images are embedded directly into the website and PDF, but files are only attached as links.

Markdown Reference

Markdown is a human-friendly, plain text publishing format (the text files are easily readable without being obfuscated by markup tags). The most useful Markdown features are shown below. Some specialist topics are explained on the advanced content page. See also the original Markdown specification and the documentation for the kramdown engine used by DocGen.

> **Note:** a page will only appear in the PDF table of contents if it has a **page heading** (see below).

Style	What you type	What it looks like

Page Heading	`Page Heading`
	`============`
		Page heading

Section Heading	`Section Heading`
	`---------------`
		Section Heading

Minor Heading	`###Minor Heading`	**Minor Heading**
False Heading (not in PDF contents list)	`<p class="false-heading">False</p>`	**False**
Emphasis (italic)	`This text has *emphasis*`	This text has *emphasis*
Strong (bold)	`This text is **bold**`	This text is **bold**
Block quotes	`> This is a block quote.`	This is a block quote
Code block (indent with tab)	`int some_code=0`	`int some_code=0`

Unordered List	`* unordered list`
	`* (items)`
		• unordered list
		• (items)

Ordered List	`1. ordered list`
	`2. (items)`
		1. ordered list
		2. (items)

External Links	`[link to Google](http://www.google.com)`	link to Google
Links to a local file	`[user guide](user_guide.pdf)`	user guide
Diagrams (embedding images)	`![](files/images/logo.png)`	DOCGEN

Simple table	`| Cell 1 | Cell 2 | Cell 3 |`
	`| Cell 4 | Cell 5 | Cell 6 |`
		Cell 1 · Cell 2 · Cell 3
		Cell 4 · Cell 5 · Cell 6

Advanced content

Overview

Any standard **HTML** can be entered in a content (.txt) file if needed (useful in cases where Markdown features are too simple).

Creating internal links to page sections

To create links to other sections within one content page, put hyphens between the words in the heading and prepend with #:

```
[link to heading](#this-is-a-heading)

... other page content here ...

This is a heading
-----------------
```

Creating internal links to other pages

To create links to *other* content pages provide the relative url to the page:

```
[link to heading](example-page.html)
```

Control of page breaks in the PDF

DocGen does not provide precision control over PDF layout. However, some steps can be taken in case of page break issues (the most common problem).

To *force* a page break, insert the following before the item you want to appear on a new page:

```
<span class="force-break"></span>
```

DocGen already tries to eliminate unsightly page breaks *inside* code blocks, block quotes, and table rows. To apply the same technique to other elements, revert to HTML and apply the **avoid-break** class. For example:

```
<p class="avoid-break">A long paragraph</p>
```

Basic Tables

An extended Markdown syntax allows basic tables to be created. Basic tables:

- have fixed width
- do not allow control of column widths
- do not provide for custom styling

What you type

```
| Heading    | Heading          | Heading |
|:-----------|:----------------:|--------:|
| Something  | Some commentary  | ?       |
| Another    | More commentary  | ?       |
| One more   | More commentary  | ?       |
```

What it looks like

| Heading | Heading | Heading | |:-----------|:----------------:|--------:| | Something | Some commentary | ? | | Another | More commentary | ? | | One more | More commentary | ? |

Custom Tables

For better control of table styling, revert to HTML. Either embed custom CSS styles in the content page (total control), or use the DocGen built-in style called **styled-table**. With **styled-table** it is possible to:

- adjust the table width and set the column widths exactly
- have zebra (alternate shaded) rows and bold columns
- have red or green text in cells, or ticks and crosses

What you type

```
<table class="styled-table" style="width:100%;">

 <tr>
  <th style="width:200px;">Heading</th>
  <th style="width:500px;">Heading</th>
  <th>Heading</th>
 </tr>

 <tr>
  <td class="bold">Example</td>
  <td>Example</td>
  <td>Example</td>
 </tr>

 <tr class="shaded">
  <td class="bold">Example</td>
  <td>Example</td>
  <td></td>
 </tr>

 <tr>
  <td class="bold">Example</td>
  <td class="green">Example</td>
  <td class="tick"></td>
 </tr>

 <tr class="shaded">
  <td class="bold">Example</td>
  <td class="red">Example</td>
  <td class="cross"></td>
 </tr>

</table>
```

What it looks like

Heading	Heading	Heading
Example	Example	Example
Example	Example	
Example	Example	
Example	Example	

Publishing with DocGen

Overview

You publish content by running **Generator.rb**. This

- **creates a static website** (each text file in Source is converter to an HTML file in Output, e.g. content.txt becomes content.html)
- **converts the whole website into a single PDF**

Before running the Generator

###Update the configuration files

These are in the 'Source' folder.

- *doc-parameters.yml* sets the attributes that will appear inside the document. Most only need to entered the first time you create a new document, **but you should make sure to always update the release field when releasing a new product version**
- *tool-behaviours.yml* allows how DocGen operates to be customised (e.g. to modify the path to 'Output' folder)

> All config files (.yml) **must** be saved with UTF-8 encoding (their default format).

###Complete table-of-contents.yml

This is also in the 'Source' folder. It sets:

1. what to call each file in the HTML table of contents and which column it should appear in.
2. which order the chapters in the PDF will be.

Note that the table of contents in the PDF is based on the page headings in your text files. The only bearing that 'table-of-contents.yml' has on the PDF is to decide the order of the chapters. You can include links to other documents or external websites in the table of contents, but they won't appear anywhere in the PDF copy.

###Update change_log.txt

If you are releasing a new version of a product, you should summarise the changes for this version and add it to the top of the page.

Run the Generator

Use the Command Prompt on Windows or the terminal on Linux / OS X and enter (substituting the correct path):

```
cd /path/to/DocGen/Generator
ruby Generator.rb
```

Advanced Usage

DocGen looks for the 'Source' files in this order:

- the sourcePath option (e.g. run `ruby Generator --sourcePath "C:/Example Source/"` noting forward slashes and quote marks)
- the working directory (only if it contains doc-parameters.yml)
- **default**: ../Source (relative to the working directory)

The DocGen Output location can be changed in tool-behaviours.yml

Review the generated content

Open **Output/documentation.html** and check the website and PDF are as desired.

Deploy your documentation

The self contained website can be placed on any web server, or used locally from a computer folder.

Troubleshooting

Why is the PDF font size or layout different when generated on Windows vs. Linux?

DocGen does not guarantee that the PDF will be precisely identical when generated on different platforms. If this matters, choose one platform and use it consistently.

Why does Chrome PDF Viewer open or print the PDF incorrectly?

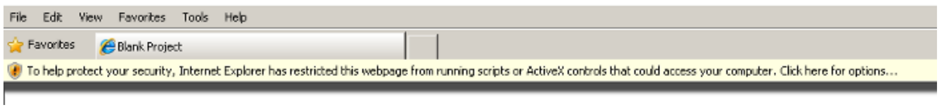
These are known issues. Only **Adobe Reader** is officially supported at this time.

Why do the links not work in the PDF?

This feature is unsupported in **wkhtmltopdf 0.9.9** but will be supported in version **0.10.0**.

Why is there an "Internet Explorer has restricted this webpage" warning?

It is a known issue that Internet Explorer prevents pages opened from a local folder from running scripts. The following warning may be seen:



To help protect your security, Internet Explorer has restricted this webpage from running scripts or ActiveX controls that could access your computer. Click here for options...

There are two solutions:

- view the pages from any web server (simply upload them)
- disable scipts in tool-behaviours.yml by setting `suppress_IE_warning: true`. This is not recommended for production use and will also disable the code syntax highlighting in Internet Explorer.

Other issues

For any other problems, please submit a an issue ticket with the following information:

- Describe the problem
- Describe the steps needed to reproduce the problem
- Attach any sample files needed to reproduce the problem

> Please always attach **log.txt** (in the Generator folder) when requesting support.

Using with Subversion

Recommended practice

When using DocGen with Subversion, the following practice is recommended (see the setup steps for help)

- **Add** the **Source** directory and its files directly to your repository (e.g. in any suitable docs directory)
- **Ignore** the **Output** directory, using Subversion properties
- Use **Subversion externals** to reference a specific release of the DocGen **Generator** and **User Guide** directories
- **Ignore log.txt** in the **Generator** directory, using Subversion properties

> Do not try to directly version the Output Folder (DocGen overwrites it). If you want to also version control the generated content files (not necessary), you must move them from the **Output** directory into another suitable folder, taking care to manually perform Subversion commands like file renames, deletions, or additions.

Setup - using DocGen in a Subversion project

i) Create a directory in your Subversion repository for DocGen documentation and perform **svn add**

ii) Apply the following Subversion property to the directory (e.g. on Windows, right click and choose **TortoiseSVN | Properties**): property - **svn:ignore**, value - **Output**

iii) Similarly, create the **svn:externals** property and set its value as follows, substituting X and A.B.C for the the desired release version from the DocGen repository tags directory:

```
^/../svnccpcommonplatformtoolbox/tags/DocGen/releases/Xx/A.B.C/Generator/ Generator
^/../svnccpcommonplatformtoolbox/tags/DocGen/releases/Xx/A.B.C/DocGen/User%20Guide/ "Generator User Guide"
```

iv) Create a child directory called **Source**, copy in the desired content files and perform **svn add**

v) Perform **svn update**

vi) Apply the following Subversion properties to the **Generator** directory: property - **svn:ignore** value - **log.txt**

Release notes

DocGen 2.0.0 released XX/XX/XXXX

- DocGen is now open source
- Rewritten in JavaScript for Node.js
- Much easier to install (hosted on npm)
- Dependencies are now version controlled (using npm)
- Modernized visual style (uses Webknife CSS framework)
- Input metadata files are now in JSON rather than YAML format
- Command-line options are now used for configuration, rather than a config file
- Command-line output is now color coded (green success, red error)
- Generating the PDF is now an optional feature
- Upgraded to the latest version of the PDF generator (wkhtmltopdf)
- Added support for list of document contributors (multiple authors)
- Added time to the generation timestamp
- Renamed 'change log' to 'release notes'
- Fixed issues with fonts and text kerning in the PDF copy
- Fixed defect where unexpected text appeared on some pages with a page table of contents
- Dropped support for Internet Explorer 7 and 8
- Dropped formal support for tool to run on multiple operating systems

Todo:

- Ability to override the version information
- Mathematical expressions with MathJax
- 'Permalinks' to headings
- Automatic secion numbering
- Automatic figure numbering
- Link checker

DocGen 1.0.1 released 18/01/2012

- Fixed a bug causing the table of contents headings to sometimes appear in the wrong order

DocGen 1.0.0 released 04/11/2011

- Initial release