



DocGen - a documentation tool

Owner: Inkit Inc.	Version: 4.2.0	Released: 15/04/2024
Author: Inkit Inc. and contributors	Contributors: Mark Macdonald	
Module:	ID:	Link: Github

Summary

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

© 2024 [Inkit Inc. and contributors](#)

MIT License. Created by DocGen 4.2.0 on 15/04/2024 at 19:42:30.

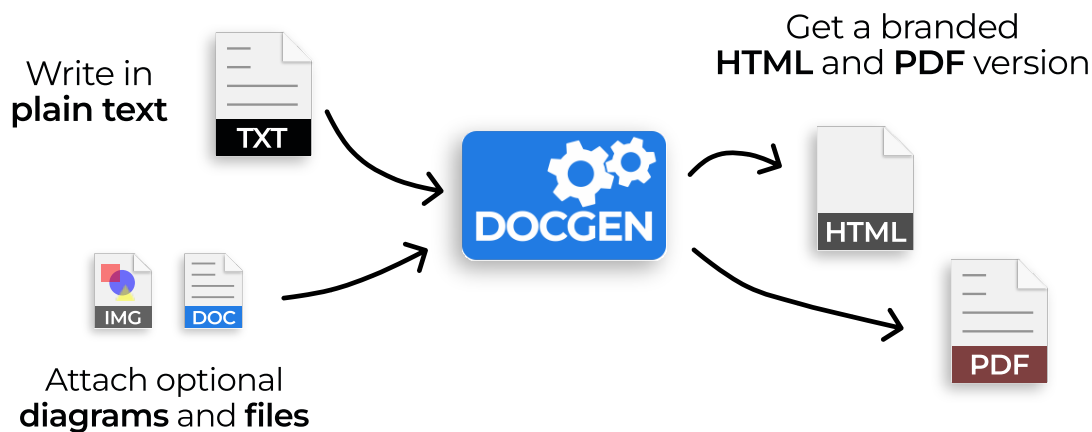
Table of Contents

Table of Contents	2
Overview	4
The Leading Open Source Documentation Tool	4
DocGen is a Static Website Generator	4
Features	4
Sponsors	5
DocGen is Sponsored By Inkit	5
How it works	6
Browser support	7
Quick start	9
Installation	10
Install Dependencies	10
Node.js and NPM	10
wkhtmltopdf (optional)	10
Install DocGen	10
Running DocGen	11
Overview	11
Command-line usage	11
Scaffold command	11
Run command	11
Writing content	13
Overview	13
Metadata	13
Plain text	14
Markdown	15
HTML	15
Embedding images	16
Icons	16
Attaching files	16
Mathematical Expressions	16
Mathematics using KaTeX	18
Mathematics using MathJax	18
Markdown reference	19
Advanced content	21
Basic tables	21
HTML tables	21
Creating internal links to page sections	21
Creating internal links to other pages	22
Control of page breaks in the PDF	22
Troubleshooting	23
Displaying detailed errors	23
PDF missing content	23
PDF layouts	23
Attached files not in PDF	23
Corrupted text characters	23
Missing logo	23
Other issues	23
Using with version control	24
Recommended practice	24
Release notes	25
DocGen 4.2.0 15/04/2024	25
DocGen 4.1.1 09/04/2024	25
DocGen 4.1.0 05/04/2024	25
DocGen 4.0.0 15/03/2024	25
DocGen 3.7.0 28/02/2024	25
DocGen 3.6.0 14/12/2023	25

DocGen 3.5.0 04/12/2023	25
DocGen 3.4.0 06/11/2023	25
DocGen 3.3.1 19/10/2023	25
DocGen 3.3.0 16/10/2023	25
DocGen 3.2.14 04/10/2023	25
DocGen 3.2.13 03/10/2023	25
DocGen 3.2.12 28/09/2023	26
DocGen 3.2.10 - 3.2.11 28/09/2023	26
DocGen 3.2.4 - 3.2.9 27/09/2023	26
DocGen 3.2.3 26/09/2023	26
DocGen 3.2.2 25/09/2023	26
DocGen 3.2.1 25/09/2023	26
DocGen 3.2.0 21/09/2023	26
DocGen 3.1.0 - 3.1.3 05/09/2023	26
DocGen 3.0.7 - 3.0.8 07/07/2023	26
DocGen 3.0.5 - 3.0.6 released 05/07/2023	26
DocGen 3.0.1 - 3.0.4 released 30/06/2023	26
DocGen 3.0.0 released 24/06/2023	27
DocGen 2.1.3 released 29/05/2015	27
DocGen 2.1.2 released 28/05/2015	27
DocGen 2.1.1 released 28/05/2015	27
DocGen 2.1.0 released 27/05/2015	27
DocGen 2.0.1 released 31/03/2015	27
DocGen 2.0.0 released 31/03/2015	27
DocGen 1.0.1 released 18/01/2012	28
DocGen 1.0.0 released 04/11/2011	28

The Leading Open Source Documentation Tool

DocGen generates HTML websites and PDF documents from plain text for free.



[Download](#)

[Report Issues](#)

DocGen is a Static Website Generator

DocGen is an open-source website generator that makes it easy to create high-quality documentation.



Self-contained website

Creates a static website that works on any server, or as local files.



Optional PDF

Also publishes the website content as a single PDF, using [wkhtmltopdf](#).



Human-friendly input

Write in plain text, or the human-friendly [Markdown](#) format.



Easy to version control

Plain text input formats work well with all version control systems.



Table of contents

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



Code syntax highlighting

Automatically highlights code blocks, using [Highlight.js](#), with language detection.



Mathematical expressions

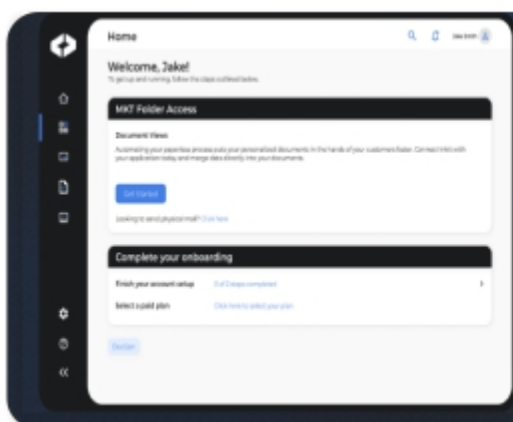
Displays mathematical expressions without plugins, using either [KaTeX](#) or [MathJax](#).



Branding and metadata

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

Sponsors



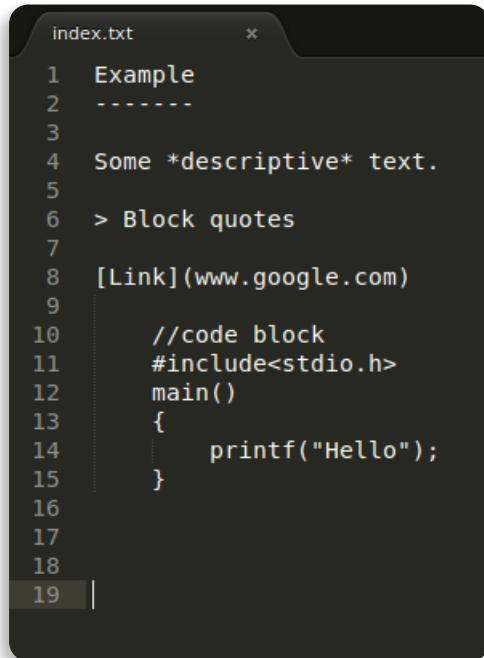
DocGen is Sponsored By Inkit

DocGen is open-source software sponsored by Inkit, the leading Zero Trust Document Generation Platform.

[Learn More](#)

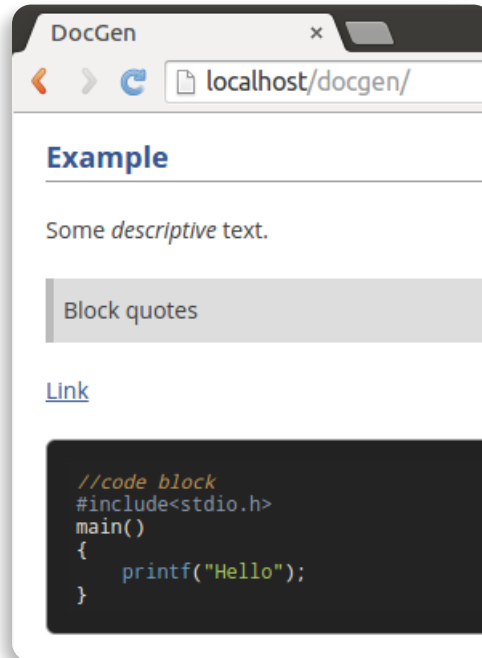
How it works

Simply [download](#) or [install](#) DocGen, and run the tool to generate websites and PDF documents.

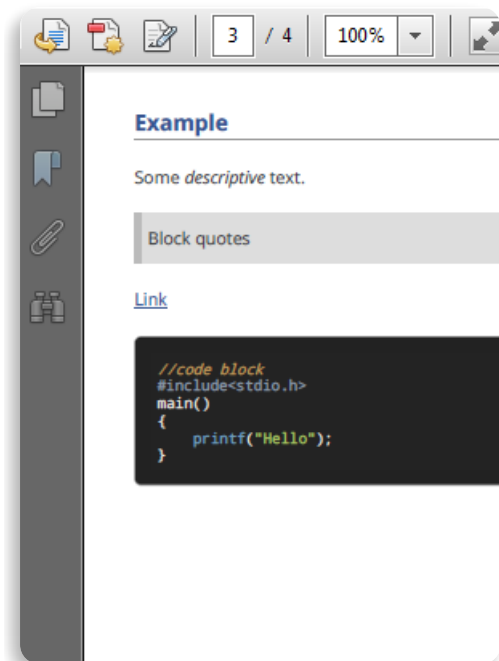


```
1 Example
2 -----
3
4 Some *descriptive* text.
5
6 > Block quotes
7
8 [Link](www.google.com)
9
10 //code block
11 #include<stdio.h>
12 main()
13 {
14     printf("Hello");
15 }
16
17
18
19
```

001 | Create content in plain text or human-friendly [Markdown](#)



002 | DocGen styles and publishes all your content as a website



003 | DocGen also creates an equivalent PDF copy

Flexible Input Formats

- Plain text
- CommonMark (Markdown)
- HTML
- LaTeX mathematical expressions
- 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)

NOTE: DocGen is intended for free-form, human-generated content which is regularly updated and improved, then automatically laid out according to a template. It is not intended as a precision PDF editing tool.

Browser support

Websites and documents generated by DocGen work in most browsers including Chrome, Edge, Firefox and Safari.



Chrome



Edge



Firefox



Safari

Quick start

In just three steps:

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

```
npm install -g docgen-tool  
docgen scaffold  
docgen run -o $HOME/docgen-example
```

Simply enter these commands in the terminal:

See the [installation guide](#) for more detailed instructions.

Installation

This section explains how to install DocGen.

NOTE - the websites and documents generated by DocGen can be used on all major operating systems and browsers without any setup or installation. These setup instructions are needed only when running the tool to generate documents.

DocGen is a command-line (CLI) tool. It runs on most major operating systems.

Install Dependencies

Before installing the tool, you need some dependencies.

Node.js and NPM

DocGen needs [Node.js](#) (JavaScript engine) and its package manager [NPM](#) which are downloaded and installed together.

See [Node.js downloads](#) and choose the installer or package for your operating system.

- **Windows:** use the [official installer](#) or [nvm-windows](#)
- **OS X:** use the [official installer](#) or [nvm](#)
- **Ubuntu Linux:** use [nvm](#) or see [other instructions](#)

wkhtmltopdf (optional)

The [wkhtmltopdf](#) package is optional and only needed if you want to enable generation of PDF documents.

To install, [download the installer](#) or package for your operating system (choose the latest version).

Install DocGen

The easiest way to install DocGen is through [npm](#) (the JavaScript package manager). Enter this terminal command:

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

DocGen can also be installed by [direct download](#).

Running DocGen

Overview

DocGen is a command-line (CLI) tool which takes plain text or markdown input files and outputs a static website. It also optionally outputs a PDF copy of the website content.

DocGen works by transforming an input directory (source files) into an output directory (website + PDF).

Command-line usage

The DocGen command-line interface includes usage help for both the tool and its subcommands:

```
docgen --help
docgen run --help
```

Scaffold command

Use the scaffold command for creating a new project. It creates an *example* input directory, by generating the minimum the skeleton input files required by DocGen. After generating them, you can customise them to create your website.

Create a scaffold template in the working directory (./):

```
docgen scaffold
```

Create a scaffold template in a specified directory:

```
docgen scaffold -o $HOME/docgen-example
```

Run command

The **run** command transforms an input directory (plain text source) into an output directory (HTML+PDF).

Basic usage:

```
docgen run -i $HOME/docgen-example -o $HOME/docgen-output
```

Optionally create a PDF:

```
docgen run -i $HOME/docgen-example -o $HOME/docgen-output -p
```

Optionally create a redirect page:

```
docgen run -i $HOME/docgen-example -o $HOME/docgen-output -r
```

The optional redirect page is an 'index.html' file that is placed in the output's parent directory. The redirect page redirects the user to the homepage in the output directory. This is mostly useful for hosting the website without having to place all the files in the root directory.

Writing content

Content for a DocGen website is authored either in:

- plain text (.txt files)
- markdown (.md files)
- HTML (embedded in .md files)

You can use any text editor or IDE to edit these. The advantage of markdown is that it will be automatically styled (e.g. headings, bullet points etc).

Image files can be embedded (via links), and other files can be attached (the website will link to these).

Additionally, some website metadata is configured via JSON files.

Overview

DocGen transforms source files from an input directory into output files in an output directory.

It takes every source file (plain text) specified in **contents.json** and converts it. Each source file becomes a separate page in the website and a separate chapter in the PDF.

DocGen adds metadata that is specified in **parameters.json**, and copies the images and files in the **files** directory to the output.

Always save input files with **UTF-8** encoding. This makes non-standard characters (ø © é etc.) work.

Metadata

parameters.json

The parameters file is used to specify metadata describing the product.

- **title** - the website title
- **name** - the website name (also used to name the PDF)
- **version** - the release version
- **date** - the release date
- **organization** - the company or organization
- **author** - the lead author of the document
- **owner** - the owner of the document
- **contributors** - list of contributors
- **website** - a link to the parent website
- **backlink** - a link back to another site (useful for integrated documentation)
- **module** - module name (useful for larger sites with submodules)
- **id** - reference number (e.g. id in a change management tool)
- **summary** - a descriptive summary of the website/document

- **marking** - license or other protective markings
- **legalese** - document markings (confidentiality, disclaimers, smallprint etc)

Values can be empty strings, but the elements are required in the JSON file.

Parameters with URLs can be either website URLs, or email addresses (specify *'mailto:name@address.com'*).

contents.json

The contents file specifies the names, locations, order, and hierarchy of the source files. It is used to generate both the web and PDF table of contents.

release-notes.md

The release notes source file is a mandatory source file (that does not need to be listed in contents.json). Use it to summarize the change history for each version of the product.

Plain text

The simplest input format is just to write in plain text. Here is an example of the source and output:

Example paragraph.

Example paragraph.

Markdown

Markdown is a human-friendly plain text markup format. The source format is easy to read and write, and the CommonMark parser translates it into HTML. DocGen uses the CommonMark standard via a package called markdown-it. Here is an example of the source and output:

Markdown Example

Paragraphs are text blocks separated by new lines.

Text can be styled: **emphasised** and ****strong****.

Here is an [example link](http://www.google.com).

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

Markdown Example

Paragraphs are text blocks separated by new lines.

Text can be styled: *emphasised* and **strong**.

Here is an example link.

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

For more examples, see the CommonMark reference.

HTML

For more complex pages not covered by CommonMark's syntax, simply use inline HTML:

<table>
<tr>
<td>Foo</td>
<td>Bar</td>
<td>Baz</td>
</tr>
</table>

Foo	Bar	Baz
-----	-----	-----

For more examples, see writing advanced content.

Inline HTML is still parsed by the CommonMark parser (HTML is allowed in CommonMark documents). In DocGen, it is also possible to bypass the CommonMark parser altogether and specify a pure HTML input page, by setting `"html": true` in a page object in *contents.json*.

Embedding images

Diagrams (in image form, e.g. JPEG, PNG, GIF etc.) should be put the *files/images* directory, and embedded as image links.

```
![[logo]](files/images/logo.svg)
```



Icons

DocGen ships with Tabler Icons built-in (a modern, open-source SVG icons library). You can insert icons in content pages by adding an HTML span tag with the `dglIcon` class, and the name of the icon in `data-name` attribute. You can also add optional classes to customize the icon styles.

Example:

```
<span class="dglIcon" data-name="graph-filled"></span>
```

You can use built-in classes to customize icon styles (small, large, info, warning, success, error).

Size:

Color:

Custom colors:

You can apply custom colors with a style attribute.

Attaching files

Other files you want to attach should go into *files* directory.

```
[attachment](user_guide.pdf)
```

attachment

Mathematical Expressions

LaTeX is the most common markup format for mathematical expressions.

Modern web browsers do not yet consistently support a common standard for authoring mathematical expressions. For this reason, extra libraries are needed. DocGen supports two widely-used web mathematics libraries out-the-box:

- **KaTeX** is fast, lightweight, and supports fewer features
- **MathJax** is slower, larger, and supports more features

The document author decides which one (or both) to use.

Mathematics using KaTeX

KaTeX is the recommended choice. It is bundled with DocGen but must be enabled by passing the `-m` option.

```
<div class="dg-katexMath">
f(x) = \int_{-\infty}^{\infty} \hat{f}(x) e^{2 \pi i x} \, dx
</div>
```

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(x) e^{2 \pi i x} \, dx$$

Mathematics using MathJax

MathJax is the fallback choice for expressions not yet supported by KaTeX. When required, MathJax can be enabled by passing the `-n` option.

DocGen uses the same MathJax configuration as the popular [Stack Exchange](#) websites.

Because MathJax is a large library, it is not bundled with DocGen, but is served from a third-party CDN (content delivery network). This means users need an active Internet connection for the MathJax feature to work.


MathJax can be slow to render. When used with the PDF feature, it may be necessary to allow more rendering time by passing the `-d [milliseconds]` option.

```
$$
f(n) = \begin{cases}
n/2, & \text{if } n \text{ is even} \\
3n+1, & \text{if } n \text{ is odd}
\end{cases}
$$
```

$$f(n) = \begin{cases} n/2, & \text{if } n \text{ is even} \\ 3n + 1, & \text{if } n \text{ is odd} \end{cases}$$

Markdown reference

This section shows examples of how to write content with [markdown](#). DocGen uses a markdown standard called CommonMark (see the [CommonMark Spec](#) for more detailed information).

Element	What you type	What it looks like in DocGen
Page Heading	# Page Heading	Page Heading
Section Heading	## Section Heading	Section Heading
Minor Heading	### Minor Heading	Minor Heading
Fake Heading (not in PDF contents list)	<p class="dg-fakeHeading">Fake</p>	Fake
Emphasis (italic)	This text has *emphasis*	This text has <i>emphasis</i>
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)	<ul style="list-style-type: none">• unordered list• (items)
Ordered List	1. ordered list 2. (items)	<ol style="list-style-type: none">1. ordered list2. (items)
External Links	[link](http://www.google.com)	link
Links to a local file	[attachment](user_guide.pdf)	attachment
Diagrams (embedding images)		

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

Basic tables

Basic tables can be inserted with the Github-flavoured Markdown [table extension](#).

```
| Heading | Heading | Heading |
|:-----:|:-----:|:-----:|
| Something | Something | ? |
| Another | Another | ? |
| One more | One more | ? |
```

Heading	Heading	Heading
Something	Something	?
Another	Another	?
One more	One more	?

HTML tables

Regular HTML can also be used for tables, allowing full custom styling (including table and column widths).

```
<table style="width:100%;">
<tr>
<th>Heading</th>
<th>Heading</th>
<th>Heading</th>
</tr>
<tr>
<td class="bold">Example</td>
<td>Example</td>
<td>Example</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td class="w-success-text">Example</td>
<td><span class="dglcon" data-name="check"></span></td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td class="w-error-text">Example</td>
<td><span class="dglcon" data-name="x"></td>
</tr>
</table>
```

Heading	Heading	Heading
Example	Example	Example
Example	Example	
Example	Example	

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 an element that should appear on a new page:

```
<p class="dg-forceBreak"></p>
```

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

```
<p class="dg-avoidBreak">A long paragraph</p>
```

Troubleshooting

This section gives help on solving common issues with DocGen.

Displaying detailed errors

Pass the **-v** (verbose) option when running DocGen to get more detailed error messages.

PDF missing content

In complex pages, the PDF generator (`wkhtmltopdf`) needs to be given enough time for dynamic content to be rendered. Pass the **-d [milliseconds]** option to increase the rendering time for each page, if required.

PDF layouts

The PDF generator `wkhtmltopdf` uses an older version of the Webkit browser engine than the engine in most modern browsers. Not all modern CSS features like grid and flexbox work, unfortunately. This only impacts the PDF output, not the website.

Attached files not in PDF

Attached files are not converted to PDF, only the web content is. The website and PDF can link to other files.

Corrupted text characters

Make sure all the input text files are saved with UTF-8 encoding.

Missing logo

The logo must be in SVG or PNG formats and saved with the path `files/images/logo.svg` or `files/images/logo.png`. It must have suitable dimensions for the header (height and width).

Other issues

For any other problems, please submit [an issue ticket](#).

Using with version control

One of the benefits of using DocGen for product software documentation is that its plain text source files are easy to version control using any modern version control tool such as [git](#).

Recommended practice

It is recommended to store the documentation **source files** (DocGen input directory) in the same version control repository as its parent project. For example, if you are using DocGen to document a software product, each release of the software app can then have a matching documentation version.

It is not necessary to version control the DocGen output, because this can always be regenerated.

Release notes

DocGen 4.2.0 15/04/2024

- better mobile-friendly styles

DocGen 4.1.1 09/04/2024

- fix Katex math rendering after removing jQuery

DocGen 4.1.0 05/04/2024

- remove dependency on jQuery

DocGen 4.0.0 15/03/2024

- new mobile-friendly responsive design with sidebar menu

DocGen 3.7.0 28/02/2024

- modernize icons (ship with [Tabler Icons](#) built-in)

DocGen 3.6.0 14/12/2023

- simplify and improve styles layer

DocGen 3.5.0 04/12/2023

- modernise view code (use React templates)

DocGen 3.4.0 06/11/2023

- modernise code (convert to TypeScript and split into modules)

DocGen 3.3.1 19/10/2023

- remove legacy rsvp package that's not needed in modern JavaScript/TypeScript

DocGen 3.3.0 16/10/2023

- start refactoring to TypeScript

DocGen 3.2.14 04/10/2023

- rebrand actions (button style updates)

DocGen 3.2.13 03/10/2023

- docs website: add Inkit sponsorship links to header

DocGen 3.2.12 28/09/2023

- fix deprecated MathJax CDN ([#77](#))

DocGen 3.2.10 - 3.2.11 28/09/2023

- improve and modernise docs (website)

DocGen 3.2.4 - 3.2.9 27/09/2023

- refactor directory structure to better suit build process
- fix path resolution error in released build

DocGen 3.2.3 26/09/2023

- fix missing build artefacts from the published NPM package ([#67](#))

DocGen 3.2.2 25/09/2023

- fix corrupted character / encoding issue in PDF footer ([#57](#))

DocGen 3.2.1 25/09/2023

- support SVG brand icons

DocGen 3.2.0 21/09/2023

- update website with rebrand and sponsorship

DocGen 3.1.0 - 3.1.3 05/09/2023

- removed dependency on Webknife CSS framework, ported styles directly to this repo

DocGen 3.0.7 - 3.0.8 07/07/2023

- automated Github CI/CD for tagging a release

DocGen 3.0.5 - 3.0.6 released 05/07/2023

- automated Github CI/CD pipeline for publishing the NPM package

DocGen 3.0.1 - 3.0.4 released 30/06/2023

- automated Github CI/CD pipeline for publishing the docs website
- modernised the build tooling (TypeScript compiler)
- publish only the dist directory to the NPM package

- better NPM commands for developing and maintaining the repository
- docs directory is now produced in a build step (build output is no longer committed)

DocGen 3.0.0 released 24/06/2023

- Ownership and copyright transferred to project sponsor [Inkit Inc](#)
- License remains open-source MIT

DocGen 2.1.3 released 29/05/2015

- Allow more protocols in CommonMark links (see [markdown-itticket #108](#))

DocGen 2.1.2 released 28/05/2015

- Fixed a regression defect (exception when running docgen scaffold) that first appeared in DocGen 2.1.0

DocGen 2.1.1 released 28/05/2015

- Upgraded all node dependencies to the latest versions
- Upgraded styles to the latest release of Webknife (1.4.0)

DocGen 2.1.0 released 27/05/2015

- Added a command-line option for specifying a custom path to wkhtmltopdf

DocGen 2.0.1 released 31/03/2015

- Fixed the node package and user guide for installing with npm install -g

DocGen 2.0.0 released 31/03/2015

- 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
- Top-level page headings are now inserted automatically (from contents.json)
- The web and PDF tables of contents both correspond to contents.json
- Command-line options are now used for configuration, rather than a config file
- Command-line output is now color coded (green success, red error)
- Added usage information to the command line interface
- Generating the PDF is now an optional feature
- Upgraded to the latest version of the PDF generator (wkhtmltopdf)
- Added support for mathematical expressions (with KaTeX or MathJax)
- Added support for a list of document contributors (for multiple authors)
- Added support for a header link back to the application (integrated docs)

- 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
- Removed support for 'Mark of the Web'

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

- Ruby implementation (not released as open source)
- Creates a static website from Markdown input files
- Also creates a PDF copy using wkhtmltopdf