



# **Technical Documentation Report**

**- a discussion of processes,  
methods and tools**

**Mark Lees** EngTech, FISTC, MIHEEM, MIET, MIPlantE, MSOE, MIDiagE  
**Technical Author**



## Contents

<b>Contents .....</b>	<b>3</b>
<b>Overview .....</b>	<b>4</b>
<b>Caveat!.....</b>	<b>4</b>
<b>1. Introduction .....</b>	<b>5</b>
<b>2. Documentation Legal/Regulatory Requirements .....</b>	<b>6</b>
<b>3. The Documentation Planning Process .....</b>	<b>9</b>
<b>4. Documentation Initial Philosophy and Approach.....</b>	<b>11</b>
<b>5. Document Content Management and Version Control .....</b>	<b>12</b>
<b>6. Documentation as Code .....</b>	<b>13</b>
<b>7. Documentation Formats and Tools .....</b>	<b>14</b>
<b>Markdown .....</b>	<b>16</b>
<b>reStructuredText.....</b>	<b>16</b>
<b>HTML / CSS / JavaScript .....</b>	<b>16</b>
<b>Sphinx .....</b>	<b>17</b>
<b>GitHub Desktop.....</b>	<b>17</b>
<b>BookStack .....</b>	<b>17</b>
<b>Pandoc .....</b>	<b>17</b>
<b>MarkdownPad 2.....</b>	<b>17</b>
<b>Gettext.....</b>	<b>18</b>
<b>8. Conclusion.....</b>	<b>19</b>

### Overview

I think it is fair to say that there is not (and probably never will be) a perfect system for producing documentation – technical or otherwise. All systems and packages have their issues or drawbacks, and the key is choosing the system from the many that exist that works best in your use-case.

In this document, I will endeavour to give my own view on the systems, procedures and methodology that I adopt in the production of technical documentation (some of which could equally be applied to non-technical documentation as well), based on my experience over almost 20 years as a technical author.

The first half of the report deals with the regulatory and other requirements, while the second half deals with the ‘mechanics’ of the documentation production and issue processes.

### Caveat!

I apologise in advance for the length of this document, but it is not possible to lay out a documentation philosophy in a single page document!

These comments are my viewpoint, which may well not fit with your viewpoint, thoughts or intent in terms of the documentation process.

# 1. Introduction

It is very important that customer-facing documentation (such as user operating manuals, service documentation, technical specifications...) meet the relevant regulatory and legal standards and requirements in the country of use. For example, all this legislation has relevance to the documentation:

## UK

- Electrical Equipment (Safety) Regulations 2016
- Supply of Machinery (Safety) Regulations 2008

## EU

- Low Voltage Directive 2014/35/EU
- Machinery Directive 2006/42/EC

## US

- 29 CFR Part 1910
- National Electrical Code (NFPA 70)

## AUS/NZ

- National Standard for Plant (NOHSC:1010[1994])

There are also several international and localised standards that also have great relevance to the content and presentation of technical documentation:

- ISO 20607
- ISO 82079
- ANSI Z535 (specifically Z535.6)

Complying with the requirements of these standards and the applicable legislation is a key part of product compliance.

Of course, none of the standards or legislation make any requirement or suggestion as to the process or tools used to produce the documentation (there are some ISO standards that do specify this, but they are outside the scope of our discussion here).

## 2. Documentation Legal/Regulatory Requirements

Content will of course vary depending on the requirement, but documentation should be fit for purpose and contain the information required (whether by law or standard), and the information should be as unambiguous as possible. As an example, this is the information required in a user operating and maintenance manual to meet legislative requirements in most locations in the world:

1	<i>PREFACE</i>
1.1	<i>Description of the User</i>
1.2	<i>Conventions Used in This Manual</i>
1.3	<i>Explanation of Safety Warnings</i>
1.4	<i>Retaining Instructions</i>
1.5	<i>Obtaining Documentation and Information</i>
1.5.1	<i>Internet</i>
1.5.2	<i>Ordering Documentation</i>
1.5.3	<i>Other languages</i>
1.5.4	<i>Documentation Feedback</i>
1.5.5	<i>Support and service</i>
2	<i>DESCRIPTION OF THE PRODUCT</i>
2.1	<i>Intended Use and Reasonably Foreseeable Misuse</i>
2.2	<i>Process Overview</i>
2.3	<i>Technical Data</i>
2.4	<i>Product Compliance</i>
2.5	<i>Product elements</i>
2.6	<i>Understanding the user interface</i>
2.7	<i>Operating Panels</i>
2.8	<i>Explanation of Auditory and Visual Signals</i>
2.9	<i>Description of Workstations</i>
2.10	<i>Lifting</i>
2.10.1	<i>Lifting accessories</i>
2.10.2	<i>Lifting machinery</i>
3	<i>SAFETY INSTRUCTIONS</i>
3.1	<i>How to Use the Product Safely</i>
3.1.1	<i>Safety information for vulnerable people</i>
3.1.2	<i>Technical life span</i>
3.1.3	<i>Safety information related to the intended use and reasonably foreseeable misuse</i>
3.1.4	<i>Personal protective Equipment</i>
3.1.5	<i>Product limitations and restrictions</i>
3.1.6	<i>Installation safety information</i>
3.1.7	<i>Safety information regarding the use</i>
3.1.8	<i>Maintenance safety information</i>
3.1.9	<i>Safe Disposal</i>
3.2	<i>Graphical Symbols</i>
3.2.1	<i>Explanation of safety information on the packaging and product</i>
3.2.2	<i>Explanation of graphical symbols in the user manual</i>
3.3	<i>Potential Health Consequences</i>
3.4	<i>Personal Protective Equipment</i>
3.5	<i>Specifications of Tools to be Used</i>
4	<i>[PROCESS/WORKFLOW 1]</i>
4.1	<i>[Sub-process/Workflow Step 1]</i>

## Technical Documentation Report

- 4.1.1 *[Procedures for step 1]*
- 4.1.2 *[Procedures for Step 2]*
- 4.2 *[Sub-process/Workflow step 2]*
- 4.2.1 *[Procedures for step 2]*
- 5 *[PREPARATION]*
- 5.1 *How to Transport and Store the Product*
- 5.1.1 *Dimensions, mass and centre of gravity*
- 5.1.2 *Lifting, handing and transporting the product*
- 5.1.3 *Storing the product*
- 5.1.4 *Storing the product during intervals in normal use*
- 5.1.5 *Securing the product against shocks*
- 5.2 *How to Install the Product*
- 5.2.1 *Removal of the transport and packaging restraints*
- 5.2.2 *Unpacking the product*
- 5.2.3 *Packaging contents*
- 5.2.4 *Minimum space needed*
- 5.2.5 *Layout plan*
- 5.2.6 *Interconnection diagram*
- 5.2.7 *Conditions for assembling*
- 5.2.8 *Installation of [product]*
- 5.2.9 *[Installation of protective features]*
- 5.2.10 *[Reduction of noise and vibrations]*
- 5.2.11 *[Stability of machinery]*
- 5.3 *[How to Commission the Product]*
- 5.3.1 *Training of operators*
- 5.3.2 *Commissioning the machinery*
- 6 *[OPERATION/USE]*
- 6.1 *[How to Use the Machinery]*
- 6.1.1 *[Operational environment]*
- 6.1.2 *[Manual operating techniques]*
- 6.1.3 *[Local/Remote operation]*
- 6.1.4 *Manual/Automatic operation*
- 6.1.5 *Vibration information for portable hand-held and hand-guided machinery*
- 6.1.6 *Portable fixing and other impact machinery*
- 6.1.7 *Starting/Stopping the product's operation*
- 6.1.8 *Checks before using the product*
- 6.2 *[How to Use the Product Remotely/Automatically]*
- 6.2.1 *[Remote use]*
- 6.2.2 *[Automatic use]*
- 6.3 *[What to Do in Emergency and Exceptional Situations]*
- 6.3.1 *[Emergency Situation]*
- 6.3.2 *[Exceptional Situations]*
- 6.4 *Optional modules*
- 6.4.1 *[Module A]*
- 6.4.2 *[Module B]*
- 7 *[MAINTENANCE]*
- 7.1 *[How to Maintain the Product]*
- 7.1.1 *[Product maintenance by non-skilled persons]*
- 7.1.2 *[Product maintenance by skilled persons]*
- 7.1.3 *[Planned maintenance of industrial plants]*
- 7.2 *How to Inspect the Product*

## Technical Documentation Report

7.2.1	<i>Weekly inspection tasks</i>
7.2.2	<i>Monthly Inspection tasks</i>
8	<i>[TROUBLESHOOTING AND REPAIR]</i>
8.1	<i>[How to Identify and Solve Problems]</i>
8.1.1	<i>[Troubleshooting and repair by non-skilled persons]</i>
8.1.2	<i>[Troubleshooting and repair by skilled persons]</i>
8.2	<i>Frequently Asked Questions</i>
8.4	<i>[How to Repair the Product]</i>
8.4.1	<i>Repair by non-skilled persons</i>
8.4.2	<i>Repair by skilled persons</i>
9	<i>DISPOSAL</i>
9.1	<i>[How to Disassemble the Product]</i>
9.2	<i>How to Recycle Parts</i>
9.3	<i>How to Dispose the Product</i>
9.3.1	<i>Disposal of electronic components</i>
9.3.2	<i>Disposal of packaging waste</i>
9.3.3	<i>Disposal of batteries</i>
10	<i>APPENDIX I – SUPPLIED ACCESSORIES, CONSUMABLES AND SPARE PARTS</i>
10.1	<i>Supplied accessories</i>
10.2	<i>Consumables</i>
10.3	<i>Spare/replacement parts</i>
11	<i>INDEX</i>
12	<i>GLOSSARY</i>
13	<i>RELATED DOCUMENTATION</i>
14	<i>DECLARATION OF CONFORMITY FOR MACHINERY</i>

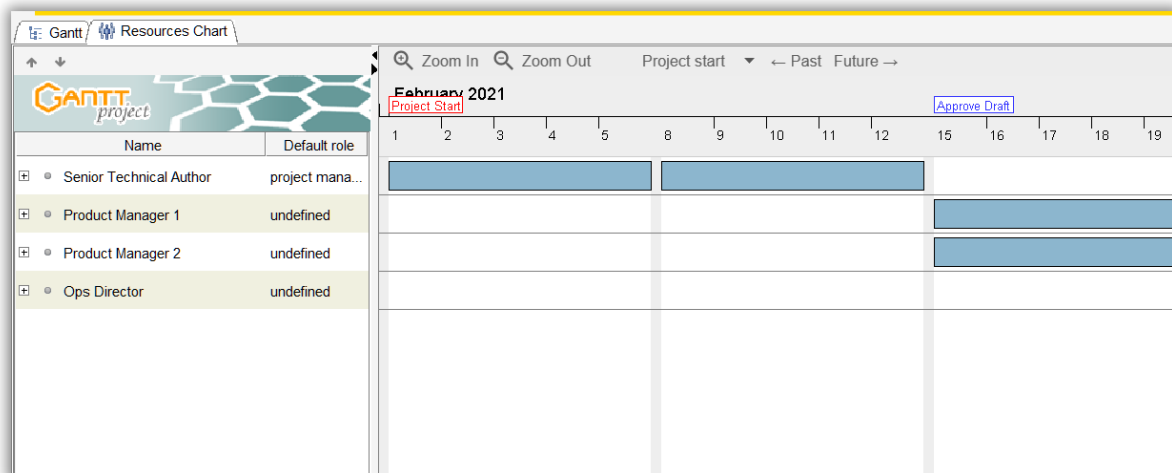
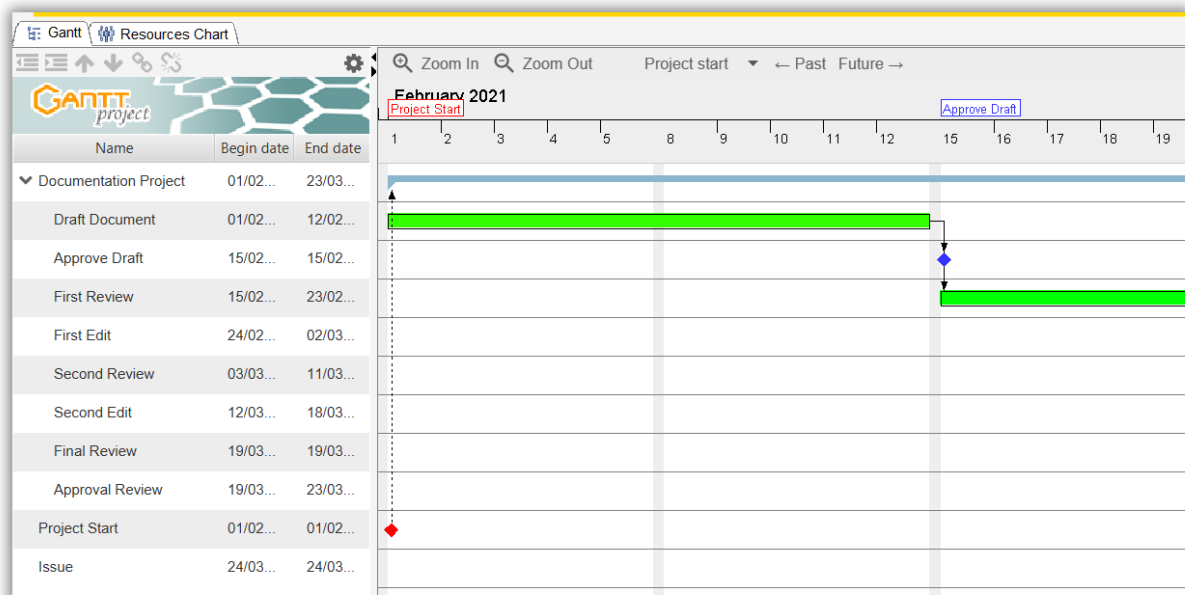
Quite a list! Not all information needs to be included for all products, but this is indicative of the content needed for most technically complex machinery or products.



## 3. The Documentation Planning Process

Before any documentation is created, the process should begin with a **project plan** that details the key stages involved along with projected timescales and milestones. The (people) resources needed for the project (subject matter experts) should be included along with an indication of when their input is required.

A typical project plan (one that I use for most of my documentation projects) looks like this:



The project plan should be updated on a regular basis throughout the lifetime of the project.

The other essential component should be a **documentation plan**. The documentation plan should also be updated throughout the project lifetime.

A typical documentation plan looks like this:

### Documentation Plan – xyz project

#### Motivation:

Documentation plans allow goals to be reached within time, cost, quality and scope constraints. The goal is to reach the target in an orderly manner. This includes the management of people assigned to tasks in the project, their availability, evaluation of their skills to perform their tasks, required training and contracting of service providers if needed.

#### Action Points:

- Schedule when things need to be ready
- Define the tools, machines and software used to produce the documentation
- Analyse what tasks need to be done to create the documentation
- Assign people to the tasks
- Define the roles in the project and specify who does what
- Analyse the risks in the project and determine what can be done to minimise them
- Specify how changes to the project are to be managed
- Specify a failure recovery plan in case something goes wrong
- Draw up a communication plan to make sure everyone finds the information they need to perform their tasks
- Specify how different versions are to be managed

#### Tips:

- Keep your documentation plan up to date and use it
- The documentation plan should be signed by everyone in the project

## 4. Documentation Initial Philosophy and Approach

After creating the documentation and project plans and ensuring legal compliance within the documentation, the philosophy or approach to take when creating documentation is straightforward and requires the use of several tools. The first (and probably the most important in terms of ensuring continuity and repeatability across all documentation) is the use of a **Style Guide**.

A style guide is simply a document that sets out how documents should look (font, size, indentation, and so on). A typical section of a style guide may look like this:

### 3. Alphabetical Listing Guide

#### 3.1. Acronyms and abbreviations

Use acronyms and abbreviations only where they will be likely to be understood by the reader. Otherwise, use the full term for the first reference, followed by the initials in brackets.

The contraction SMS should only be used in internal documentation circulated within the company; in any documentation that will be issued externally, the company name should be used in full.

Use full stops after abbreviations to indicate missing letters, e.g. no. 56, but not after contractions, acronyms or initialisms. For example: Mr, dept, ISBN, TAFE.

Please note that the term 'documentation' applies to documents in any format, and the style guide can be used not just for conventional documents (e.g., Microsoft Word) but also electronic documents (e.g., online/web based), and in fact should contain the styles and rules used for all formats of documentation.

Once the style guide has been created, the next step is to create a set of document templates that follow the requirements laid out in the style guide – this saves the user from having to think about how to lay out a document, as the framework has already been created for them. The intent should be to adopt a reusable content approach when producing documents (more about this later in this report).

Once a consistent set of templates are created, steps need to be taken to ensure no confidential/hidden content is included, and that it contains both the creation date of the document and a suitable number that identifies the document template or document itself – this functionality can be integrated into the template in a straightforward way so that again the user does not need to worry about this.

Having templates and documents then leads on to ensuring that document numbering/identification is performed in a logical and sequential way – there are many ways to achieve this, but further detail is outside the scope of this report.

## 5. Document Content Management and Version Control

Once we have our multitude of documents, we need to manage them in two ways:

- Document content
- Documents themselves

The document content is generally managed by the tool used to produce it, and typically this is in most cases, Microsoft Word. Although Word is a fine tool for producing some content, it falls far short as a content management tool and in revision control – it just does not have this functionality. This is where we need to be using a suitable **CMS** (Content Management System). There are many CMS systems available, and all have their advantages and drawbacks. The use of a CMS is in my opinion essential, and my preference is to use a CMS called BookStack.

BookStack is open source (so it is free to use), gives great control of a document format and appearance, can use templates and is simple to learn and use (it uses the analogy of a library – lots of shelves that can contain books that in turn contain chapters and pages. It also has many advanced features (including Markdown support, drawing support and a free text search system), and includes a form of revision control. However, there are of course other means of content management as well

While the revision control within the CMS provides an easy way to see the audit trail of content change (as well as also being able to see the changes), this will only work for content that is created within the CMS. Any documentation that is produced using a different method will need to use a different form of version control.

I often use **GitHub** as a method of version control - and you can also use it as a form of content management as well. GitHub supports the creation of content using markdown (in the same way that BookStack does). Although git was originally developed for software projects, it does lend itself to documentation very well, particularly if a documentation as code philosophy is adopted (see the next section for more details of this).

The generally recommended workflow for documentation projects using GitHub is:

- A master branch that the next release of your product documentation is developed on
- Git branches for ongoing maintenance of each version of your documentation that is maintained
- Git tags for specific released versions that users might be using

An example:

- The master branch is your 2.2 version that isn't released yet
- 2.1.X is your release branch for the 2.1 version
- 2.1.1 is a similar tag of your 2.1 branch, with the latest release
- 2.1.0 is the first tag of your 2.1 branch

## 6. Documentation as Code

Documentation as Code (generally referred to as 'Docs as Code') refers to a philosophy that you should be writing documentation with the same tools that developers use to create code, for example:

- Issue Trackers
- Version Control (Git / GitHub)
- Plain Text Markup (Markdown, reStructuredText, AsciiDoc)
- Documentation Reviews
- Automated Tests

This approach can enable a culture where the author and the contributing parties (subject matter experts, etc.) all feel they have ownership of the documentation, and work together to make the documentation as good and usable as possible.

Using plain-text markup for the documentation content gives lightweight content that can be ported to many different applications (as opposed to binary formats such as Word documents that are difficult to reuse outside of their original application).

It has been argued that this approach should not be adopted, because content isn't code; I always counter this with the fact that code can be content! Levity aside, there are many benefits to adopting this approach in the production and issue of documentation.

## 7. Documentation Formats and Tools

There is a myriad of software tools that can be used to produce documentation, and every individual has their own preferred packages. I of course am no different in this regard, but whenever possible my preference is to use open-source software and lightweight formats that allow easy reuse of generated content as well as localisation.

Although applications such as Microsoft Word and Acrobat (as two examples) are useful tools, they produce their content in a proprietary, binary format that can result in extremely large file sizes and be prone to becoming corrupted – resulting in you not being able to open the document

Compare these two screenshots. The first is a Microsoft Word document viewed in a standard text editor:



```

PKK  ! Bx0LZ  [Content_Types].xml  (
1"%3:p3V4fNÜ5l  µw%è=-"i7+Üxä-d&á"0pAÉ6€14%L600#µÄ'fs
OææfX0 Ž*•v$z3ü3ä:~p-p)0µ  ^
"2x5}nH"ndÜsÓXg•L,,`%è|éÖY"-P0rÜfs0ðŽ?~PWŽ it4Q+ÉÆ"|wa0-|T\y¹°H,NÜ0ä0Ux%´úÚ-D/0'fÜš4X;Ýžÿ( '|00%<EäÜ00)'à0 ;,
00_rels/.rels  (
@ifÿfN%QÜÄ0EN/c0Ü0Ü  [IL0Ü0jxþy<000]éagÉ00"0zsæFuà"]00-U
5%  0ü^Ä[ÜXx  •...X¥1x0pâ
>æöfÿÊ#I)Éf<Y0Sÿ00'0~`ÍÄ0â*D0â$
i")Í0c$3£žqUx÷~3  "100jH[{  =E%†00i0~
f?±-3-00Äp²]ÄTê",20j)0,0l0/%æ'b-
00%Ñ0ez£$Ä...,  i  %/ü|f\0Zpç$æ0?6i!Y´_áo0æ]A00  ýÿ0 PK000  ! qí
üw  Ä4  word/document.xmli>Ürâ,0†iS•w`)-%>00€±$S,a0é™´$Ýé!|(Éb™[´"eIÜ!o~'  ÄE
eÉ'1É•00...) Ä0ff0  Épâç†$0iÄ0(02ör  ?€00!A6Ž0ÜÉÄ¿$S
>0ÉÉ0C~æ¥áâà1,  ?^ýü0?İ†â,~%azIR"-t0<,0L«*0èz0LÄÄ/?$QPde60>0Y4g"Í0,,ú<+Æ:000¿0"
Ä²"i9~zİ-fV.x0Mm\0SY  0~`ÜE0>,5âp"†nê|S00  ${`à!0p[šÉÊ
!r00`jCÉ8LÉ%INÄ"0;0;L  o"ñÄ"6pJ60I00•..."-HüJ&<[=ñ<>Y~!...s¿ŠFQ0U0R00DNÆ00»0, 'µ0
0iÄ0$0†10w*Üâ`VHÄ0pÄ4%2}00o/]0b-p7UÜ680=x<0-%È0r0â<0Ž00&
50Éý0NÜ'qwb<†;N-çÄ"Ü.r), <üý",±|"0;0'X00Ä,,06;K0Iá²áf\³ä\,c  é00†
Ä00~$Ä[
=XİP¥0i85:0ft"N't,Ü1Ž00Y0(Äý$0İŽ01VéQ~ü:10²Y%T^S00q0,,æj;°†V<ÿê",_gİf0İe1J,ác04+üQ,-'0k'G000W0«00?Ä  M0,Ä•Ü%0²ñE0æ²,
s¿0?Jr
Äæc0PçÊà_0\äz000*#s†r«4pr9  r00´-Yx...Éd~ZŽ•ÉtÄ%?<«Í'ë-ÜSëB]†00;0Ä%-0  0è*³hÉFubä"0µü]pxİC080ÜVpİ"ëyz[w_è00ZÜ&0İti
é0...0-ü0²07Y$00Pn-0s9bp-É0Ä]gÜ]W00«%m00e00%"µ  JA~>Z0:Y<K0İ•+i2ê[0iİ0Üÿ.Rßš0lûj%2uY%•w0N0š0Nün->S°Tè*V<ñY#f...0J™)00ë|:
µuü0Ü'Äu^0.@
0»€<
*0İ%<00Ži'wf0ó±00L`ÇÄýâÄÄ0<0ÿ0•İrk%Ä,U0Ä0z0SNµ{ÄÄfDİ  vWYİ3)D[f0.0~90µÄfêÜñšÜ00-B|{šý]"µ^0-äÈ&0x-CTÄEÄ0X=i\Y$pa`ê0'
00ÄièÄ0p\³ÿtæ0>3`0ß|íC?¥0V0~pw12™0m£x0z!`šE  0è{&óÄ>€00E0<péàD`zDâ^}0

```

Although Word and other applications can now produce documentation using OpenXML formats, XML is extremely difficult to decipher, and the document is still saved in a binary format!

## Technical Documentation Report

This is the same document, but created in Markdown and again viewed in a standard text editor:



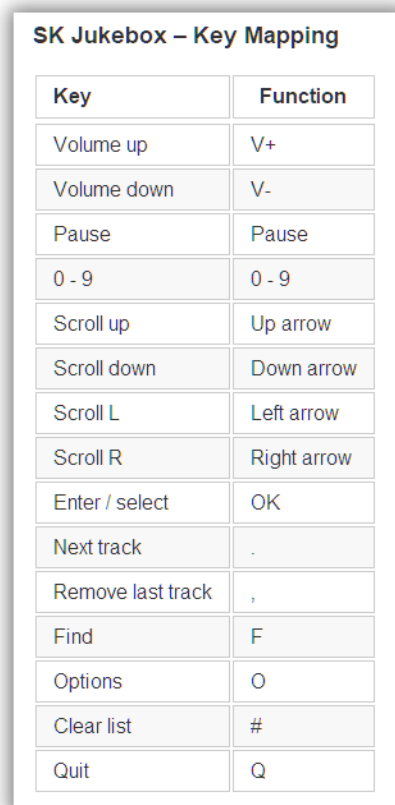
The screenshot shows a Notepad window titled "SK Jukebox - Notepad". The menu bar includes "File", "Edit", and "View". The document content is as follows:

```
### SK Jukebox - Key Mapping

**Key** | **Function**
:--- | ---
Volume up | V+
Volume down | V-
Pause | Pause
0 - 9 | 0 - 9
Scroll up | Up arrow
Scroll down | Down arrow
Scroll L | Left arrow
Scroll R | Right arrow
Enter / select | OK
Next track | .
Remove last track | ,
Find | F
Options | O
Clear list | #
Quit | Q
```

The status bar at the bottom indicates "Ln 1, Col 1", "100%", "Windows (CRLF)", and "UTF-8".

Although the formatting looks strange, the document is perfectly readable, and could be copied into another application easily. This is what the Markdown document looks like when processed and viewed in a web browser:



Key	Function
Volume up	V+
Volume down	V-
Pause	Pause
0 - 9	0 - 9
Scroll up	Up arrow
Scroll down	Down arrow
Scroll L	Left arrow
Scroll R	Right arrow
Enter / select	OK
Next track	.
Remove last track	,
Find	F
Options	O
Clear list	#
Quit	Q

## Technical Documentation Report

My preference is to (whenever possible) produce documentation that is portable, reusable and intended for viewing in any web browser (Chrome, Firefox, Opera, Safari...), eBook viewer (such as EPUBReader) or PDF viewer (Acrobat Reader or web browser plug-ins), for either online or offline viewing.

As such, this requires a specific set of tools as I mentioned earlier. All the tools and packages I use are open source or very low cost and give me the maximum flexibility in how I produce my documentation. I feel extremely strongly that documentation should move away from the view that it is intended to be printed – electronic document formats such as HTML and PDF are now widely accepted and used (and web-based content is not optimised for print). However, it is relatively easy to produce lightweight content that also scales well to a print format.

I have listed below the formats I prefer to use when creating documentation

### Markdown

Markdown is a lightweight markup language that you can use to add formatting elements to plaintext documents. Created by John Gruber in 2004, Markdown is now one of the world's most popular markup languages. Its syntax is like HTML but has a much more limited set of features. The overriding design goal for Markdown's formatting syntax is to make it as readable as possible.

The idea is that a Markdown-formatted document should be publishable as-is, as plain text, without looking like it's been marked up with tags or formatting instructions. Markdown is easy to learn and use and produces well-presented web content.

**My preferred authoring applications:** Sublime Text; MarkdownPad 2; Notepad ++

### reStructuredText

reStructuredText is an easy-to-read plaintext markup syntax and parser system. It is useful for in-line documentation (such as for code), for quickly creating web pages, and for standalone documents. reStructuredText is designed for extensibility.

The intended purpose of the markup is the conversion of reStructuredText documents into useful structured data formats such as HTML, ePub or PDF

**My preferred authoring application:** Sublime Text

### HTML / CSS / JavaScript

All three are well-known methods of creating web-based content; however, I rarely create content this way anymore – I prefer to use Markdown or reStructuredText and let the authoring tool parser create the output content.

**My preferred authoring application:** Notepad ++



## Technical Documentation Report

I have mentioned some of the preferred applications I use to create content, but there are also some tools I use to parse that content and produce the required output. These are detailed below:

### Sphinx

Sphinx is a powerful documentation generator that has many features for writing technical documentation including:

- Generates web pages, printable PDFs, documents for e-readers (ePub), and more all from the same sources
- Uses reStructuredText or Markdown for creating content
- Has an extensive system of cross-referencing code and documentation
- Has an in-built search engine that does not require a connection to the internet

Sphinx is known as an **SSG** (static site generator) and makes it easy to create intelligent and well-structured documentation that can be used on or offline. It natively produces output in a variety of formats:

- HTML
- Windows HTML Help)
- LaTeX (for printable PDF versions)
- ePub
- Texinfo

### GitHub Desktop

Git has a very steep learning curve, particularly as it is a command-line tool. GitHub Desktop is a GUI based application that makes the whole process of creating repos (repositories) for documentation or code very simple. Managing and controlling documentation (through adding branches or just by editing locally) is very straightforward, and it is possible to see exactly what changes have been made through a colour-coding system.

### BookStack

BookStack is a simple, self-hosted, easy-to-use Content Management platform for organising and storing information. Please refer to the description of a CMS earlier in this report.

### Pandoc

Pandoc is a command-line tool that understands many plaintext, markup and binary formats and can convert files from one markup format to another (including Microsoft Word files!). Pandoc currently supports over 50 input/output file formats.

### MarkdownPad 2

MarkdownPad 2 is a GUI based tool for creating HTML or PDF output from Markdown content. The application has a built-in parser to generate the HTML or PDF files. It also allows for real-time previews of the parsed content as well as support for several 'flavours' of Markdown (including GitHub Flavoured Markdown).

### Gettext

Gettext is a translation tool used for localisation. It effectively splits the reStructuredText or Markdown content into line-by-line splits consisting of two strings in each split (the original text, then an empty string). These files can then be sent to a translation company who can translate the text and fill in the empty string with the translation. This makes reusing the translations very easy.

## 8. Conclusion

It has been my intention in this report to highlight the various methodologies and processes to use when producing technical documentation.

I look forward to your comments on the points that I have documented here.



63 St Mary Axe  
London  
EC3A 8AA

0203 003 5086