Getting started with AsciiDoctor

Start writing, styling, building, and previewing content with AsciiDoctor.

Basic markup reference

As a technical writer you should be familiar with basic AsciiDoc syntax. The following section shows a few of the most common examples of AsciiDoc markup usage. For a more in-depth look at AsciiDoc syntax, check out the AsciiDoctor syntax quick reference.

Headings

Create headings by stacking = symbols at the beginning of a line. There 5 heading levels, numbered 0 to 4. Your modules should start with a level 0 heading.

```
= This is a level 0 heading
===== This is a level 4 heading
```

Block ID

Add block IDs to your sections to make internal linking easier.

Block IDs:

- must be unique to each section
- must not contain spaces

```
[id='this-is-a-block-id']
= This is a section heading
```

Paragraph

You don't need to use special markup to type a paragraph in AsciiDoc. You can separate paragraphs by placing an empty line between them.

Like this!

Inline markup

Bold

- **Constrained**

```
*Constrained*
```

- **Un**constrained

```
**Un**constrained
```

Italics

- *Constrained*

```
_Constrained_
```

- *Un*constrained

```
__Un__constrained
```

Monospace

- `Constrained`

```
`Unconstrained`
```

- `Un`constrained

```
``Un``constrained
```

  Use monospace to mark up literal values, such as:

  - file names
  - shell commands
  - names of services
  - variable names and variable values

You can combine inline markup to access additional styling options.

Lists

Unordered

An unordered list makes long enumerations:

- clear
- readable
- easy to follow

Always place an empty line before the first item of an unordered list. Use up to 5 asterisks to create different level nested sub-items.

```
This is the list title:

* Item 1
** Sub-item 1
* Item 2
* Item 3
```

Ordered

Use an ordered list to describe a sequence of steps:

1. Learn AsciiDoc
2. Write some documentation
3. Publish your documentation
4. Be awesome

```
This is the list title:

. First step
.. Sub-step
... Sub-sub-step
. Second step
. Third step
```

Ensure that you place an empty line before the first item of an ordered list. Use up to 5 periods to create different level nested list sub-items.

AsciiDoc also provides markup for other, more specialized list types.

Block markup

Code Block

Use code blocks to show longer sections of code or multi-line commands. Set the `[source]` attribute and specify the language of the code to enable syntax highlighting:

Source

```
[source,bash]
----
#!/bin/bash
file='book.txt'
while read line; do
echo $line
done < $file
----
```

Rendered

```
#!/bin/bash
file='book.txt'
while read line; do
echo $line
done < $file
```

Literal block

Use literal blocks to show terminal output examples or the listed contents of a file:

Source

```
....
$ systemctl status
● hostname.foo
    State: degraded
     Jobs: 0 queued
   Failed: 1 units
    Since: Sun 2019-05-05 20:40:03 CEST; 4h 11min ago
   CGroup: /
           ├─user.slice
           │ └─user-1000.slice
           │   ├─user@1000.service
           │   │ ├─gvfs-goa-volume-monitor.service
           │   │ │ └─2930 /usr/libexec/gvfs-goa-volume-monitor
           │   │ ├─xdg-permission-store.service
....
```

Rendered

```
$ systemctl status
● hostname.foo
    State: degraded
     Jobs: 0 queued
   Failed: 1 units
    Since: Sun 2019-05-05 20:40:03 CEST; 4h 11min ago
   CGroup: /
           ├─user.slice
           │ └─user-1000.slice
           │   ├─user@1000.service
           │   │ ├─gvfs-goa-volume-monitor.service
           │   │ │ └─2930 /usr/libexec/gvfs-goa-volume-monitor
           │   │ ├─xdg-permission-store.service....
...
```

Links

External

Create a link to an external resource:

```
link:https://www.example.com[link text]
```

The following link, for example, takes you to the AsciiDoctor Writer's Guide

Relative

Link to a section in the same document using the section ID. This also works for included AsciiDoc content.

```
xref:section-id[link text]
```

Complex content

Use the + character to group content of different types together. This ensures that combined content is correctly indented, and preserves the numbering of the steps. The following example features a list with codes examples in two of its steps:

Source

```
. Download `myLatestProject.tar.gz`
. Extract the project files:
+
[source,bash]
----
$ tar -xzvf myLatestProject.tar.gz
----
+
. Install the files:
+
[source,bash]
----
./install.sh
----
```

Rendered

1. Download `myLatestProject.tar.gz`
2. Extract the project files:

   ```
   $ tar -xzvf myLatestProject.tar.gz
   ```

3. Install the files:

   ```
   ./install.sh
   ```

Including content form other files

AsciiDoc allows you to include other files in your title, Use this functionality to compose modules into assemblies and write documentation in a modular way.

Use the `include::` directive to include content from an external resource in your title. Specify the location of the included module relative to file that you are including it in. Use the `leveloffset` attribute to ensure that the included pieces nest correctly. For example:

```
include::modules/_included-file-name_.adoc[leveloffset=+1]
```

Building content

You should have AsciiDoctor installed on your system. You can use it to build your content.

1. Ensure that you are in the directory that contains your main `.adoc` file and execute the following command:.

```
$ asciidoctor <em>main_file_name</em>.adoc
```

The main file is titled `master.adoc` by default, although you can choose a different name.

Previewing rendered content

To view the built content, open the `main_file_name.html` file in a browser.