Here's how the node docs work.

1:1 relationship from `lib/<module>.js` to `doc/api/<module>.md`.

Each type of heading has a description block.

module

<!--introduced_in=v0.10.0-->

> Stability: 2 - Stable

A description and examples.

module.property
<!-- YAML
added: v0.10.0
-->

* {type}

A description of the property.

module.someFunction(x, y, [z=100])
<!-- YAML
added: v0.10.0
-->

* `x` {string} The description of the string.
* `y` {boolean} Should I stay or should I go?
* `z` {number} How many zebras to bring. **Default:** `100`.

A description of the function.

module.someNewFunction(x)
<!-- YAML
added: REPLACEME
-->

* `x` {string} The description of the string.

This feature is not in a release yet.

Event: 'blerg'
<!-- YAML
added: v0.10.0
-->

* `anArg` {type} A description of the listener argument.

Modules don't usually raise events on themselves. `cluster` is the
only exception.

Class: SomeClass
<!-- YAML
added: v0.10.0
-->

A description of the class.

SomeClass.classMethod(anArg)
<!-- YAML
added: v0.10.0
-->

* `anArg` {Object} Just an argument.
  * `field` {string} `anArg` can have this field.
  * `field2` {boolean} Another field. **Default:** `false`.
* Returns: {boolean} `true` if it worked.

A description of the method for humans.

SomeClass.nextSibling()
<!-- YAML
added: v0.10.0
-->

* Returns: {SomeClass | null} The next `SomeClass` in line.

`SomeClass` must be registered in `tools/doc/type-parser.mjs`
to be properly parsed in `{type}` fields.

SomeClass.someProperty
<!-- YAML
added: v0.10.0
-->

* {string}

The indication of what `someProperty` is.

Event: 'grelb'
<!-- YAML

```
added: v0.10.0
-->
```

* `isBlerg` {boolean}

This event is emitted on instances of `SomeClass`, not on the module itself.

- Classes have (description, Properties, Methods, Events).
- Events have (list of listener arguments, description).
- Functions have (list of arguments, returned value if defined, description).
- Methods have (list of arguments, returned value if defined, description).
- Modules have (description, Properties, Functions, Classes, Examples).
- Properties have (type, description).