Complete idiot’s introduction to yaml:

YAML is a very simple, text/human-readable annotation format that can be used to store data. It is natively supported by Bukkit and almost every plugin uses it. CraftIRC's configuration file, **config.yml**, uses it; It's important that you do not break the formatting while you are editing it or the plugin won't run.

Programmers deal with much more complicated stuff without blinking twice, so if you really want to run a server don't be a lazy jackass and make an effort to understand this. It will be useful knowledge for *every* plugin!

**Basic principles to remember while working with config.yml:**

* Always save using the UTF-8 encoding to minimize the possibility of errors.
* Never, EVER use the TAB character anywhere in it.
* Editing with a WYSIWYG text processor, such as Microsoft Word, is not recommended.
* Use a monowidth/monospaced font to view the contents. Examples you may have in your computer: Fixedsys, Lucida Console, Consolas, Monaco, DejaVu Sans Mono, Courier New.

Whenever you see the **#** character anywhere in the file that isn't first enclosed in '' (quotes) or "" (double quotes), it marks the beginning of a **comment**. This means all the text after it, up until the end of the line, is *completely ignored* and has no effect. You can use this to write notes on the file or temporarily disable one or more options.

These are all comments:

#debug: false

#Hello there, how are you?

debug: false # <- The line is read up to this character!

This is not commented:

name: '#changeme'

But this is!:

#name: '#changeme'

**Key-value pairs**

YAML keeps data stored as a map containing keys and values associated to those keys. This map is in no particular order, so you can reorder it at will. Each pair is in the format KEY: VALUE. For example:

minecraft-tag: 'minecraft'

cancelled-tag: 'cancelled'

Note the 'quotes' around the value. When the value is a text string, we use the quotes to make sure any special characters aren't given special meaning, and instead are all kept as part of the value. So even though they are optional, using them is highly recommended.

Other than text, numbers and true/false, the value associated to a key can also be another map of key-value pairs. To achieve this, omit the value and instead write the key-value pairs in the following lines. However, you must prefix them with at least one more SPACE character than the key.

YAML will consider that lines prefixed with more spaces than the parent key are contained inside it; Moreover, all lines must be prefixed with the same amount of spaces to belong to the same map. So this works:

formatting:

from-game:

chat: '(%sender%) %message%'

action: '\* %sender% %message%'

This works too:

formatting:

from-game:

chat: '(%sender%) %message%'

action: '\* %sender% %message%'

But this doesn't work:

formatting:

from-game:

chat: '(%sender%) %message%'

action: '\* %sender% %message%'

**Alternative format**

YAML supports an alternative syntax to store key-value maps, useful for compressing small maps into a single line. The syntax is: {KEY: VALUE, KEY: VALUE, ...}. The above example would become:

formatting: {from-game: {chat: '(%sender%) %message%', action: '\* %sender% %message%'}}

**Lists**

There is one other data structure from YAML we need to use - The list. Lists are used to store a collection of ordered values. The values are not associated with a key, only with a positional index obtained from the order in which they are specified (item 1, item 2, etc.).

mylist:

- 'item 1'

- 'item 2'

Like key-value pairs, list items are defined in the lines below the list key, all with the same amount of spaces prefixing them (at least as many spaces as the parent key). The difference is that they begin with a dash (**-**). Here's another valid list:

mylist:

- 100

- 200

Now, remember how you could put key-value maps inside key-value maps? You can do the same with lists. You can have:

* Maps inside maps (as seen in the previous section)
* Lists inside maps (as seen immediately above - lists are, by default, inside maps, since they need a key)
* Maps inside lists

channels:

- name: '#mychannel'

password: ''

- name: '#myprivatechannel'

password: 'mypassword'

* Lists inside lists (not currently used in CraftIRC)

twobytwotable:

- - 'a1'

- 'a2'

- - 'b1'

- 'b2'

**Alternative format**

Guess what, there's also a one-line format for lists! It is [ITEM1, ITEM2, ITEM3, ...]. Here's the above example:

twobytwotable: [ ['a1', 'a2'], ['b1', 'b2'] ]

**Conclusion**

Now that you're a more knowledgeable person, I hope you can add new items to a list or keep your key-value pairs aligned without making a scene. Everything you see in the CraftIRC config.yml will be either maps or lists of the types described above. You can freely adjust your config file to your taste by swapping to/from the alternative formats, changing the order of the keys in a map and adding/removing comments.