Keeping your source simple

With good content comes great presentation

Isaac "FishTank" Fischer, Community Technical Specialist

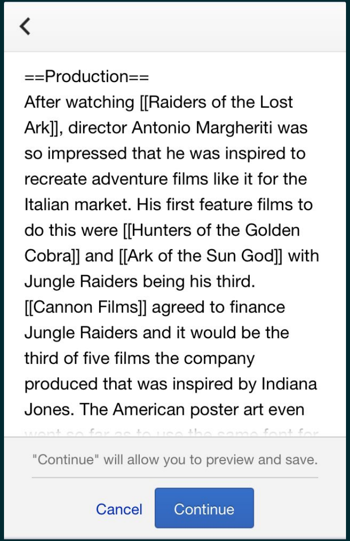
FANDOM

If you primarily use the Visual Editor, this presentation will not make much sense to you. Mastery of the Source Editor allows for more control and detail, but is not intended for casual editors.

Mobile editing

On FANDOM communities, it is possible to edit articles with mobile devices. The (typically) small screen size and lack of a full keyboard make some editing tasks challenging.

Whether you're editing on desktop or on mobile, there are some benefits to keeping your source code simple.



==Production==
After watching [[Raiders of the Lost Ark]], director Antonio Margheriti was so impressed that he was inspired to recreate adventure films like it for the Italian market. His first feature films to do this were [[Hunters of the Golden Cobra]] and [[Ark of the Sun God]] with Jungle Raiders being his third. [[Cannon Films]] agreed to finance Jungle Raiders and it would be the third of five films the company produced that was inspired by Indiana Jones. The American poster art even went so far as to use the same font for

"Continue" will allow you to preview and save.

PRO-TIP: Many (non-article) pages (like templates) can not be edited (or even viewed) using Mercury. Mobile display can be simulated, in some cases, by using the Mobile Preview option.

Wikitext versus HTML

Wikitext is designed for simplicity

Syntax developments like {{curvy {brackets}}} were seen as a way to make basic text formatting easier to understand for new editors.

Wikitext calls (or invokes) other wikitext in levels (usually templates inside templates).

All wikitext can be converted to HTML.

HTML is designed for flexibility

Nesting <tags><inside /></tags> use the full power of the Web, even if the syntax is complex and not as easy to understand.

HTML does not usually embed inside other HTML, so it's usually on one level.

Not all HTML has a wikitext equivalent.

PRO-TIP: Some functions and extensions use <angular> brackets, but are not actually HTML. These "tag extensions" and {{#ParserFunctions}} run software code on the server before displaying it.

Coding Conventions

Sticking to an agreed-upon set of rules on how your source code is spaced, indented, and organized is called following coding conventions. While it often doesn't make a difference in how the page or template is displayed, it may make it easier for others to understand.

Many source code habits come from the mindset that space is at a premium. In this era of computing, space in source code is not expensive. If the code is easier to read in a more organized way, that makes it easier to edit.

```
{{Character | name = Batman | home = Gotham City |
occupation = crime-fighting vigilante}}
```

```
{{Character
| name = Batman
| home = Gotham City
| occupation = crime-fighting vigilante}}
```

PRO-TIP: The mobile page editor in Mercury edits the source of a single section of an article at a time. Spacing and indentation make a big difference in how easy it is to edit on mobile devices.

Design

For designs to work on a mobile screen, they either have to be responsive (ie. able to adjust and reflow their shape for a wide or narrow display) or they should be able to be scrolled with a fingertip.

Many block elements (usually tables) with a large fixed width will have to scroll on Mercury to see the whole element.

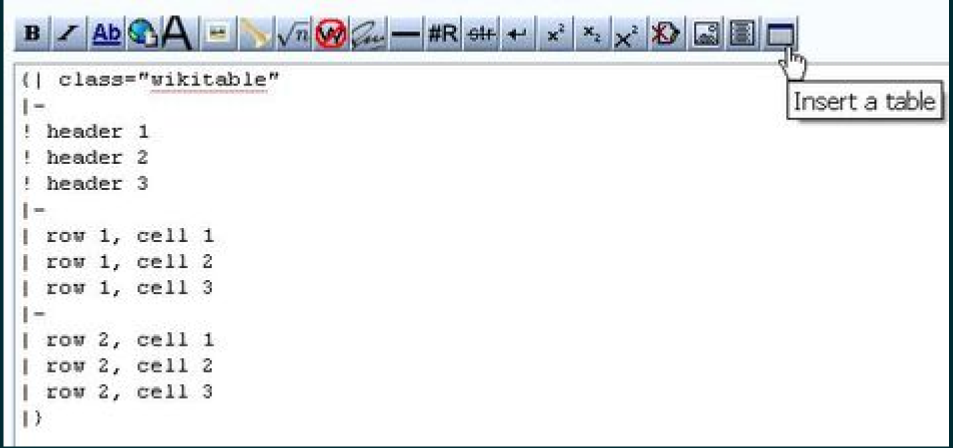
To keep your source code simple (and your content flexible) it may be easier to break up large or wide tables.

PRO-TIP: Plenty of beautiful infoboxes are wide at the top of articles. Most of them do not reduce gracefully to mobile display.

Tables

Tables are a great way to organize spreadsheet-like data, whether they are constructed using wikitext or HTML. They're not necessarily a great way to make notices and whole pages. Tables made with relatively simple layouts scroll well with a finger swipe, and are zebra-striped in Mercury for clarity.

(No colspan / rowspan, no nesting tables, and no building rows or cells with templates (contents only).)



```
{| class="wikitable"
|-
! header 1
! header 2
! header 3
|-
| row 1, cell 1
| row 1, cell 2
| row 1, cell 3
|-
| row 2, cell 1
| row 2, cell 2
| row 2, cell 3
|}
```

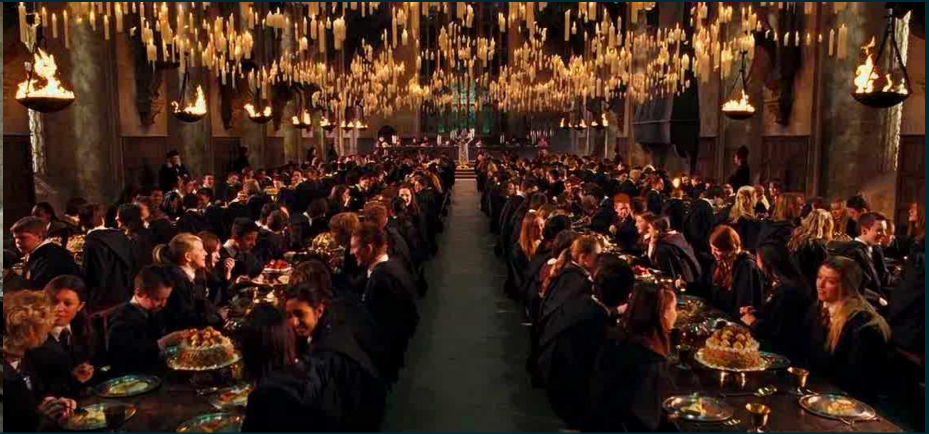
```
<table class="wikitable">
<tr><th>header 1</th>
    <th>header 2</th>
    <th>header 3</th></tr>

<tr><td>row 1, cell 1</td>
    <td>row 1, cell 2</td>
    <td>row 1, cell
3</td></tr>

<tr><td>row 2, cell 1</td>
    <td>row 2, cell 2</td>
    <td>row 2, cell
3</td></tr>
</table>
```

PRO-TIP: Tables are an exception to the "limit your inline CSS" recommendation, and rows and cells can bear individual style="" declarations. Custom CSS classes make templates less portable.

Organized Tables



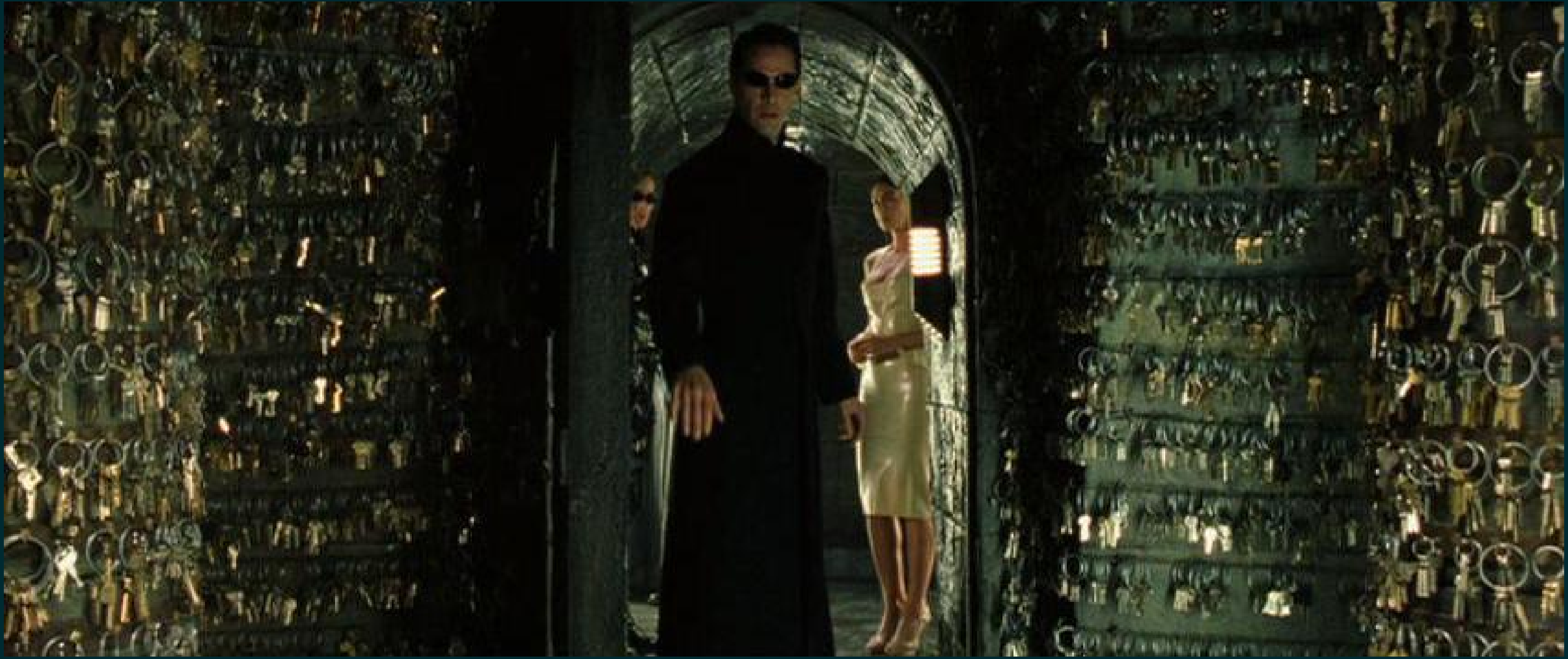
Intuitive Templating

It's already challenging for new editors to get used to the rules and guidelines editing on a new community. Templates that are very complicated, with un-intuitive parameter names, are enough to scare users away.

`| date_of_birth =` is fairly simple, but `| dob =` (and other acronyms) may not be understood by everyone who edits. Unless your template is well documented (and even then, it should be an intuitive and clear format), only your community veterans will know that `|G2_atk_r` is "Game 2's attack reduction stat" and that it should be a number only. There's no penalty for spelling out full words.

PRO-TIP: There's no harm in adding a description to your short infobox labels, either.
`<label><abbr title="Hit Points">HP</abbr></label>` goes a long way towards understanding.

Intuitive Templating



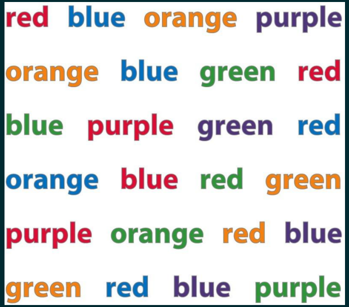
PRO-TIP: Always document your templates for the benefit of new editors. If you're the only one who knows where to find the right key, it doesn't do much good for the community in the long term.

Inline CSS

The easiest way to make your source simpler is to avoid inline CSS.

Keeping this in mind, most CSS is ignored in Mercury. Classic wikitext (like '''**bold**''' and ''*italic*'') works perfectly in Mercury and is easier to read than <b>**bold**</b>, <strong>**bold**</strong>, or <span style="font-weight: 700>**bold**</span> (which all mean the same thing).

If you use CSS for color, remember that "white" is a lot easier to understand than #FFFFFF.



red blue orange purple

orange blue green red

blue purple green red

orange blue red green

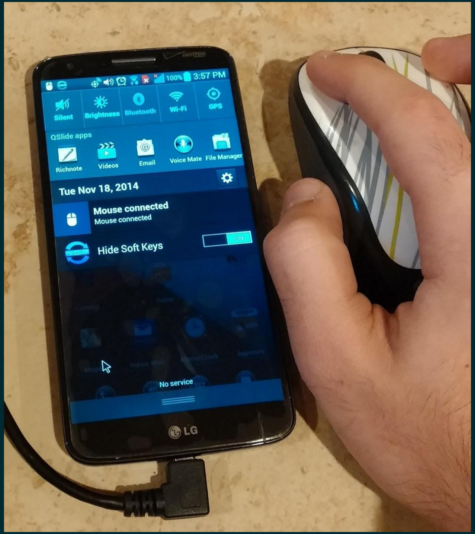
purple orange red blue

green red blue purple

PRO-TIP: The <span> HTML tag for pieces of text is usually pretty safe for Mercury (since text flows). Using <div> may not be. If you've added a <div> and inline style to an article, be sure to preview the mobile result.

JavaScript and Interactivity

Content modified (or created) using JavaScript and jQuery will not be visible on Mercury. Content that requires a mouse-over or hover (like tooltips) are not effective on devices that don't have pointers (like touchscreens). Therefore content should be expanded where it makes sense, because "expanded" is the only way to read it in Mercury.

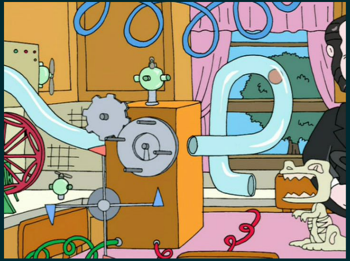


PRO-TIP: If what's on your page can only be made with interactive JavaScript (like a game stats calculator), consider that it might not be considered "content" at all, and where this "tool" or "feature" might fit in another namespace.

Lua and nested templates

Lua is an alternative to wikitext for making complex templates. It's made more like a programming language, and because making actual Lua functions (stored in a page called a "Module") is not simple to learn or use, we don't promote it often at FANDOM.

That said, using Lua for discrete, self-contained functions can produce much more readable and understandable code at the template level than using templates inside templates inside templates or ParserFunctions building on each other.



If you know what a subroutine is and how to use one, Lua may make your code simpler. Be prepared to help others learn it also, so that you are not the only one doing maintenance.

PRO-TIP: The FANDOM Open Source Library (aka dev.wikia.com) has excellent information and resources about using Lua. It should be your first stop for any Lua-related project.

Questions?