An Emacs-based writing workflow inspired by TEX and WEB, targeting the Web

Christian Gagné

Abstract

I present here a practical method developed with colleagues working in the humanities, whereby they produce content using macro-less notations (such as Markdown), which I integrate and publish on the Web by using macro-rich notations such as Emacs Org-mode and TEX. Over time, this method's general applicability has caused me to entertain a notational pipe dream: a minimalistic substitution syntax, suitable for content work in any field (technical or otherwise), which would yield both TEX on the one hand, and on the other hand HTML and XML styled with TEX-equivalent CSS.

1 Introduction

When using Emacs, there are at least four different things called 'macros' — and I have come to use all four in my work as a content integrator: Emacs Lisp macros, keyboard macros recorded in a non-Lisp expression language, Org-mode lexical macros and TEX macros written with AUC-TEX. The Org lexical macros turn out to be especially useful for one who is used to TEX and needs to produce HTML: they bring some of the power of writing in `WEB` to the Web.

The following concerns a methodology developed over the course of my work as a research and teaching assistant in a literature department, with influence from my concurrent work as a teaching assistant in computer science, hence the mix of developer-friendly and user-friendly solutions.

Some background will be relevant in order to explain my methodological choices: during my philosophy studies (which have both continental and analytic components), I was lured into a Medieval Studies research unit and became their "IT guy". At that point I was already a confirmed TEX user and my new medieval interest involved taming the XML beast. I happily obliged; however, the road ended back at a distinctly TEX-flavored abode.

My first job was to teach my literary colleagues to write in a format other than Word or LibreOffice. Given that I love the interplay of theory and practice, I organized workshops for them as I pursued my gradual discovery of theoretical computer science, philosophy of language and of what Edward Tufte calls *analytic design* [11].

It was also in the same school year that I began using Emacs in earnest, printing out reference cards and practicing proper buffer movement. What led me to Emacs was first AUC-TeX, then Org. With AUC-TeX, I first realized how much the dynamics of writing matter. Here I was teaching my colleagues to write Markdown and TEI XML by hand, just as I had begun to discover the joys of writing with macros! However, when I tried to introduce this into their workflow, it seemed to them one step too far: they said they preferred writing XML by hand, and were intimidated by transformations and macro expansions of any kind. Such is one of the reasons behind the hybrid workflow we adopted for our research unit's Web site [6].

While I studied the many alternatives for implementing the TEI guidelines and transforming TEI into other formats, I came across the name Sebastian Rahtz many times. His texts, code examples and generally remarkable implication in such matters set important precedents and helped me bridge the gap between the TeX and XML mindsets. As such, I am very thankful for the pioneering work of Sebastian Rahtz and others in the field of semantic transcription. Below, I will refer to an example of the way I interpreted the *Text Encoding Initiative Guidelines* [2]. I have gotten much mileage out of the Guidelines, but in an unorthodox way.

As it turns out, the way I have used TEI ties in beautifully with some recent wide-ranging efforts in the Web design community to integrate solid information architecture into the use of markup, in order to make it both meaningful and aesthetically pleasing. In other words, I believe that some of those very same 'semantic markup' techniques used in an academic setting will be relevant for Web content work in any field, be it technical, scientific or creative.

2 Flexible delimiters for Web content work

Org's markup syntax is very rich and parses into intricate structures. One of my favorite stated principles from the syntax specification [9] is that 'the paragraph is the unit of measurement'. An order of magnitude higher, many nesting constructs exist. For example, lines beginning with stars do make trees, properly speaking:

```
* Section title example in Org

** This is a subheading --
   and a node in the tree
```

Markdown and Org share this ability to create implicit tree structures through the use of headings. This is applicable in many situations. However, for my scenario I needed to name my content blocks using native HTML5 vocabulary and I could not commit to regular section titles that would convert to `h1` ... `h6` elements. The reason was that I had to make content blocks that would nest arbitrarily: they had to be *closed under inclusion*, one might say (though Web developers rarely speak this way). Web people call this working *content-out* instead of *canvas-in*, after the expressions consecrated by information architects' discussions at Web sites such as *A List Apart* (`alistapart.com`) and *Boxes and Arrows* (`boxesandarrows.com`). The following example is typical of what was needed on our Web site:

```
#+begin_section
#+attr_html: :id about-blocks
             :class mt_sectitle
#+begin_header
How block-delimited lines become paragraphs,
which are at the syntactic level
of mt_concept(elements) in Org parlance
#+end_header

The parsing rules dictate that, because Org
is very much a mt_emphasis(line-based) format
(in its surface guise), paragraphs are created
inside the special blocks as expected.

As for the lexical macros, they are expanded
at the very beginning of the export process.
#+end_section
```

What do we have in this Org example? Two nested special blocks which will be converted to the appropriate HTML5 or LaTeX constructs, as appropriate, and also some Org lexical macros similar to those of the C preprocessor, m4 or WEB. Here the lexical macros are shown in their fontified form, a feature added in recent Org versions. Under the fontification hood, there are actually three pairs of braces around the whole macro invocation! The block is much more readable when the braces are removed by fontification, and the macro objects are also syntax-highlighted.

Org is line-based, contrary to token-based macro engines such as m4. In this, Org is closer to Markdown, in which blank lines are a staple of the syntax. In both Org and Markdown then, the document is delimited into blocks and inlines [8]. This is important for ergonomics, since it reveals document structure graphically. This is also one of TeX's strong points, and probably one of the many reasons why TeX is often deemed easier to learn than the SGML family. In fact, though my colleagues preferred writing their TEI XML by hand, I allowed them to write all other content types in Markdown, freely interspersed with HTML as they needed. Upon reception, I processed their documents with Pandoc, then *refactored* them, so to speak, in Emacs. Whenever I wished to integrate multiple complex sources, Org was my intermediate language of choice.

The `begin` ... `end` pairs in Org have extra power, making them invaluable for producing SGML-style markup: they can conditionally write attributes in the resulting element tags. This can be used to add *classes* for CSS selection, identifiers, target URI's, microformats or even Resource Description Framework statements. In the above example, `id` and `class` attributes will be added to the `header` block upon HTML5 export. If the export target is LaTeX, simple environments will be produced that bear the names of the Org special blocks. This affords one the occasion to write some sophisticated definitions for said environments, perhaps going as far as to reproduce a given CSS layout. It is still an open question for me what exact set of packages and commands one would need to adequately reproduce some of the more idiomatic CSS stylings, for example `relative`, `absolute` and `fixed` element positioning — and the true holy grail consists of that plus a TeX implementation of *Flexbox*!

The above example is used in a Web demo [4], with some typeset examples, which acts as a companion piece to the present article.

3 Applying WEB-style substitutions to Web content

The constructs nested inside the element blocks, namely the macro objects, have an `mt_` prefix meaning 'multi-target', since the macros define different substitutions for different target formats. In the accompanying Web demo, examples are given for an HTML rendition, plus an SVG rendition produced with a TeX engine and an SVG converter.

The initial motivation for publishing SVG text blocks typeset with TeX comes from the Medieval Studies project: I wanted to produce facsimile excerpts of the manuscript transcriptions. I found that the best way to reproduce the beautiful hand-written text was to procure a historically-accurate font with many alternates and make a proof-of-concept example by manually adding OpenType substitutions. This proof-of-concept facsimile is available at [3]. The XeTeX engine was used with the `standalone` class and the `fontspec` package. Because the Web browsers cannot at present be trusted with such complex typography, I used the Poppler tools to convert the PDF to SVG with all text converted to paths.

In [1, p. 45], Robert Bringhurst mentions the rich textures that Renaissance typographers achieved with a single type size and hand-drawn additions. This same 'sensuous evenness of texture' is very much present already in fifteenth-century manuscripts such as those reproduced on our Web site, and which the early Renaissance typesetters duly imitated. In fact, this has a very material basis in the tools of calligraphy, for the broad-nib pen that dominated European writing until the Early Modern period had a more or less fixed width, depending on the pressure applied by the scribe. This width yields a fundamental tone, a stroke that serves as the basis for a scale. With the appropriate stroke modulation techniques, it allows the scribe to play within friendly neighboring scales, but the relative evenness of texture is to be seen as a blessing.

I strongly believe in the power of harmony and counterpoint in all media, so I always seek such effects, including with modern type. That is why I speak of my workflow as generally applicable to Web content work in any field, as long as the contributors care about aesthetics and believe in the dignity of the craft. In my facsimile, I have attempted to approximate the scribe's creative freedom by applying many OpenType substitutions, which recovers some of the rich graphical harmonics created by a steady and energetic hand. After many facsimile pieces have been created, one could create appropriate macros to group OpenType substitutions, thus creating a mini-macro package giving the flavor of a particular scribe's craft!

I must stress that the TEI documents referred to are very much fragments and are not validated in any way. In order to ease my colleagues into the workflow, I decided to treat anything they produced as relevant and to act upon the principle that they had good reasons for using whatever structures they had written. This remained scalable because only a dozen node types were allowed, including both elements and attributes. From the beginning, I had planned to write a Relax NG grammar when the process stabilized, but the production of this grammar remains an open ticket to this day. This has turned out to be a blessing, since it allowed my colleagues to inform the vocabulary themselves according to their needs, without any external pressure from a *pesky grammarian*. Again, the process was established from the bottom up, and this has turned out to be a boon. The graphical fidelity of the transcriptions is all the better for it: they are actually *readable*, which is a noteworthy milestone in and of itself.

In his presentation of the `WEB` literate programming system, Donald Knuth discussed how he chose to make his lexical macros as simple as possible, with only one parameter allowed, in a section entitled 'Occam's Razor' [7, p. 121]. The elegant simplicity of the approach summarized there is one of my chief inspirations in developing the present workflow and in forming my conviction concerning the general applicability of substitution in the act of writing. What is more, The Occam with whom originates that saying, being one of the greatest logicians of the Late Middle Ages, is most appropriately mentioned here, since it is with him and some contemporaries that the foundations were laid for a mathematical consideration of language items as discrete structures that can be manipulated algorithmically, as in the substitution techniques discussed here. I mention this in passing, o readers, knowing all too well that defending this statement properly would lead us down a much longer road which we will have to travel another time.

4 Equational inspiration

I have shown how the many aspects of macro substitution have shaped my experience of writing technical documents. Another theoretical development has led me to further consider macro-based writing as a most appropriate way of writing. This year, I have been introduced to equational logic through a computer science course given in my department. This course is based on the Gries-Schneider approach [5], which is in turn influenced by Edsger Dijkstra among others — and Leslie Lamport adopts a similar approach in his work. I read the course's introductory material and learned that equality is taken as the most fundamental relation and is *defined in terms of substitution*. Finding this epistemologically pleasing, I asked around whether substitution and the Leibniz rule, so defined, functioned as a theoretical base for macros in Lisp or TEX, or rather whether macros were effectively applications of the substitution principle, to which the reply was that, given the definitions we were handling, this was certainly the case. I then thought, with my usual inclination towards holistic conclusions, that surely this was yet another sign that macro languages are most appropriate to the very nature of *writing with a computer*.

5 Conclusion

In the end, even though I am presenting an Emacs-based writing workflow, what I am truly committed to is the epistemology of writing that emerges from all this and the algorithmic principles that make it possible in practice. That is why I am still trying to find better notations for writers. Recently, this has meant Web writers especially, as I strongly wish to see the riches of TEX-style composition being distributed liberally among as many people as possible, be they front-end developers, engineers, blog writers or designers. That is the meaning of my syntax essay at the end of the companion demo page [4]: reflecting upon ways to make macro-based writing more alluring to people who would traditionally never have thought of using Emacs or TEX. In a word, my wish is for Web writing to become more *organic*, both technically and culturally.

References

[1] Robert Bringhurst. *The Elements of Typographic Style*. Hartley & Marks, 2005.

[2] TEI Consortium. *Text Encoding Initiative Guidelines*. Oct. 5, 2015. `http://www.tei-c.org/Guidelines`.

[3] Christian Gagné. *Miroer du monde: comparaison et mises en relation entre BnF fr. 684 et BnF fr. 328*. 2016–. `http://hu15.github.io/histoires-universelles-xv/miroir-du-monde/comp/comp_fr684-fr328.xhtml`.

[4] Christian Gagné. *Organic TEX Demo*. Aug. 2016. `https://waidanian.github.io/organic-tex-demo`.

[5] David Gries and Fred B. Schneider. *Calculational Logic*. `http://www.cs.cornell.edu/gries/Logic/intro.html`.

[6] Groupe de recherche HU15. *(H)istoires (U)niverselles 15*. 2016–. `http://hu15.github.io/histoires-universelles-xv`.

[7] Donald E. Knuth. Literate Programming (1984). *Literate Programming*. Center for the Study of Language and Information, 1992, pp. 99–136.

[8] John MacFarlane. *CommonMark Spec*. July 15, 2016. `http://spec.commonmark.org/0.26/#blocks-and-inlines`.

[9] Org Dev Team. *Org Syntax (draft)*. Apr. 6, 2016. `http://orgmode.org/worg/dev/org-syntax.html`.

[10] David Walden. "Macro memories, 1964–2013". *TUGboat* 35:1 (2014), pp. 99–110. `http://tug.org/TUGboat/tb35-1/tb109walden.pdf`.

[11] Mark Zachry and Charlotte Thralls. "An Interview with Edward R. Tufte". *Technical Communication Quarterly* 13:4 (2004), pp. 447–462. `https://www.edwardtufte.com/tufte/s15427625tcq1304_5.pdf`.

◇ Christian Gagné
`christiangagne (at) outlook dot com`
`https://waidanian.wordpress.com`