

Re: Future of LISP. Alternative to XML. Web 3.0?

Source: <http://coding.derkeiler.com/Archive/Lisp/comp.lang.lisp/2006-12/msg00962.html>

- *From:* "Juan R." <juanrgonzaleza@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* 8 Dec 2006 07:42:06 -0800
-

Robert Maas, see <http://tinyurl.com/uh3t> wrote:

Just now, in preparation for responding to your posting. It has some rough edges. For example, when I saw `<carriage-return>` I didn't realize it meant literally the cr character within the message, I thought the textual string `<carriage-return>` literally appeared, and I hoped it would be later explained *which* carriage returns (later) would be ignored and when the ignoring of them would be turned back off. Then the examples just had nothing at all, just a new-line situation, with no explanation what was supposed to be there. Apparently what's there is the option of the local implementation, and regardless of which of the four choices applies, a backslant at the end of the line causes that line to be directly adjacent to the start of the next line ignoring whichever of the four types of newline happened to be present. That ought to be explained a little, instead of leaving it to the reader to guess what is meant. Also the BNF has semicolons at the ends of some lines but not others, and I have no idea what those semicolons denote. I'm pretty sure they aren't supposed to be literally present in the syntax being defined?

Why do not ask him directly? <http://theory.lcs.mit.edu/~rivest/>

S-expr are more concise and easy to parse than XML. You would note improvements when dealing with high volume traffic sites or manipulating large files.

That's not the point I was making. If somebody implements a server that merely duplicates an already-available service, except for using s-expressions instead of XML, nobody is going to use it, because it's cheaper to keep the existing XML software and continue paying for extra bandwidth rather than rewrite all the software just to save a fraction on the bandwidth.

Re: Future of LISP. Alternative to XML. Web 3.0?

This is a case-by-case. It depends of the cost of the XML software being used and of the cost of the extra bandwidth and/or server disk and processing time.

In [1] you can find an example of s-expr instead XML for military data. He claims reduction in a factor of 4.

Others in newsgroup could offer you more help and examples that i can.

We need a brand-new service that somebody might want to use, which isn't already provided via XML, so somebody would have incentive play with it manually to get a feel for the service offered and to *see* how incredibly more legible and significantly more efficient it is compared to XML, and then having seen that they might be willing to download our new s-expression plugin for distributed-computing API so that their software could then use XML or s-expressions equally. Note that our server could provide output in three modes, fully prettyprinted, typical-compact prettyprinted, and ultra-compact, making it easy to switch between debugging/explanatory mode and maximal-efficient mode.

Something somewhat similar i am trying to do for CanonML for scientific and mathematical data.

For instance, something like $[a / 2]$ is left at editing and storing mode (you call "ultra-compact"). At this representation one typically saves 10x over typical XML encoding. Next, it can be parsed to $[\text{\#num } a] [\text{\#den } 2]$ for 'prettyprinted'.

Current browsers do not understand this code, but is easily parsed to XML-MAIDEN format [2] or to HTML format and next displayed via standard CSS. The program is to continuum substitution of XML, HTML, CSS layers for new CanonML layers.

As an alternative to a brand-new service, one hack I have been thinking about from time to time is to get permission to use Google's XML interface to their search engine, then write an interface that maps between s-expressions and XML, so somebody could connect to my CGI application, submit Google searches in s-expression format, and get back RMI objects/references in s-expression format, and then browse the RMI objects remotely via my service. As a distributed application, for my own personal use, I could write an s-expression client that submitted a Google search via my s-expression interface, then browsed the resultset to organize it better, then finally presents the nicely re-organized index of search results as a tree of virtual Web pages for me to browse from a regular Web browser.

Re: Future of LISP. Alternative to XML. Web 3.0?

I have in mind something similar except i would not convert to xml format and that i would use cnml expressions as input. When working, it would do in any search engine, currently i am researching on the surface of google, Yahoo!, and MSN engines.

I asked Google representative for specific support of metadata on their scholar optimized engine but they recomend to me PDF format only for online data, therein i am trying to develop a new search engine service can access to metadata otherwise would be lost.

Do *any* of those LISP projects have a (HTTP-accessible) server I can play with, to see their various encodings of XML, to compare them?

Online search by SXML, DSSSL, GSXML, Latte, Scribe... LML, the HOP project nayone cited here other day, and you can find useful information about the s-expressions being used and lisp and scheme sources to final html and xml docs togheter with apps and code for both off and online conversions. DSSSL is a ISO standard. SXML approach is very 'complete' including SSAX (Scheme) parsers, libraries, and others...

There is too many information.

Do you really mean modification (incompatible changes), or merely enhancements/additions?

Both. At Arc community you can find both people adding cosmetic changes over a CL or Scheme basis and people going beyond.

3) Documentation. XML was designed for documents, S-expr for data.

But documents *are* (one kind of) data!

Yes, by document vs. data people mean mixed vs. pure data.

S-expressions have already been used to represent lots of different kinds of data. Surely they can also represent documents.

Yes. You can also represent docs in stone. The point is that s-expr are not optimal for documents, somewhat as XML is not optimal for data.

Re: Future of LISP. Alternative to XML. Web 3.0?

There are several documentation systems based in s-expr (e.g. Scribe), but failed to be popular when compared with acceptance of HTML or TeX.

Both HTML and TeX suck when it comes to expressing mathematics.

Yes, but people has found both TeX and HTML more comfortable than s-expr for documentation purposes during decades. Scribe is even previous to SGML!

Compare these two ways of expressing the same thing:

(expt (arrayelement x 5) 2)

x superscript 2 backspace subscript 5 endelement

Can you see that s-expressions clearly show the nesting of mathematical notation, whereas TeX acts like a manual typewriter moving head up or down with no reference to what the mathematics is supposed to really mean?

2

x

5

Is that even possible to express in HTML?

In a tricky way? Yes. I use a cnml format next is converted to HTML at `_client_` for rendering. Using classes and storing the source in a title or alt element, you can maintain semantics: `title='(expt (arrayelement x 5) 2)'`

Of course, you cannot access to the information in a easy way and information is not structured, therein tricky. At least i can offer today more mathematical information is possible with TeX and MathML approaches and information. Full approach would need of a XML browser or a CanonML or LISP browser.

By the way, way back in 1975 I wrote a crude text-layout program (MRPP3 a.k.a. POX, successor to something (MRPPP and MRPP2) I had written in 1969-70), which had overlays (composable pieces of picture which could then be combined in various geometric arrangements to make larger pictures), and macros (syntax vaguely like Algol/MacSyma when called, defined usually to build overlays out of pieces). Bill Gosper wrote a dandy set of macros for his own use, enabling him to use my program to lay out immense continued fractions and matrices and hypergeometric series etc., not as pretty font-wise as Knuth's carefully tuned Tex, but properly structured per the nested mathematical expressions.

Re: Future of LISP. Alternative to XML. Web 3.0?

Later, using PSL (Portable Standard Lisp), I re-wrote the program (now MRPP4) to work directly from Lisp nested-lists, basically using the same internal representation as Tony Hearn's "Reduce", producing output in several formats (ASCII-art, and IBM CRTs attached to VM/CMS system, and Unix workstations using X), including a visual interactive structure editor where the cursor covered an entire sub-expression, shown as that part of the display highlighted, and could be zoomed in or out and panned up or down the current level of structure, and then a command could be executed on the current sub-expression as embedded in the larger expression to perform algebraic manipulations.

MacSyma used a vaguely similar internal representation to Reduce, a bit more ugly in my opinion. But in any case, the basic idea is that internal Lisp structures were used to represent mathematical documents, and then of course PRINT could externalize such data and READ could get stuff back in, and MRPP4 could render the data structures in the usual mathematical-notation format. (MacSyma also had a math-format display mode, which it normally used for all output in the ordinary Read-MathEvaluate-Print loop. AFAIK MacSyma display output provided *only* ASCII-art mode, but I'm probably mistaken.)

Wow!

Anyway, one application I'm thinking of writing is a port of MRPP4 to CGI/CMUCL, whereby s-expressions from the client get internalized to Lisp structures and then displayed as MRPP4/MacSyma-style ASCII art, using `<pre>...</pre>` of course to make sure things line up in all browsers.

Interesting! I decided do not use ASCII art. I use basic CSS+HTML rendering techniques i adapted from XML-Maiden project to HTML, working minimally for 'modern' Gecko, trident, and Opera engines (> 95% clients). Very old browsers can still access the math but displaying the source code. I work stuff as $[Y = [a / 2]]$ but techniques can be easily adapted to $(= Y (/ a 2))$.

Of course I'd also have an output mode for box-diagrams showing how the internal data is structured. The math-display output would work only for valid math structures, whereas the box-diagrams would work for all Lisp data.

Interesting again, i plan an automatic tree drawing and also an first class typesetting engine without TeX metrics not conventions for a future.

Re: Future of LISP. Alternative to XML. Web 3.0?

For instance $[a = [b + c]]$ i.e. $(= a (+ b c))$. TeX does not correctly structure the maths, then the engine simply generates a sequence of tokens $a = b + c$ and next recognizes the $+$ as special binary operator and the $=$ as special relation operator, checks some built-in metric tables and adds automatic $4/18$ and $5/18$ spacing around operators respectively.

This is time-consuming, obligate to store the metric table and assumes meaning about symbols, e.g. assumes that $=$ is operator equals when could be not. Presentation MathML correct this and can differentiate between $\langle mi \rangle = \langle /mi \rangle$ and $\langle mo \rangle = \langle /mo \rangle$.

I had thought of something as direct rendering of underlying nested structure $[a = [b + c]]$ where each box $[]$ receive extra space around it. This way the b got more space around the $=$ than around the $+$. It is automatic, display represents the nested structure, is cheap and do not need of metric tables or predefined symbols. What do you think?

Indeed! With 100 proposed, I would expect to see live Web-accessible demos of at least twenty of them already. Do you know of any such?

Yes, I revised all of that in previous research. Several focus on some subset as XSLT or XHTML. Other try to be alternative as data format others focus on some specific point as non-hierarchies.

Most of them are accessible online and contain tutorials, sources... Some of them are non-continued projects as Latte. Wikipedia works with alternative to xhtml, YAML got popularity, last changes on LMNL were presented in last Extreme markup conference (there is an online demo in some place now I do not remember) Markdown is used in several blogs... Please follow the links or Google a bit.

SXML provides a representation for XML documents in the Scheme programming language – a kind of dialect of LISP–. Is that really true, presumably some specialized Scheme notation that is incompatible with Common Lisp?

I do not understand you, is true what?

I reserved the sequence `::` for the tags because `is "never" used in real practice.`

The `::` notation is not more used switching to `#`, but this is not still

Re: Future of LISP. Alternative to XML. Web 3.0?

closed issue. The character is escaped when appearing.

At one point you said that [...] is more legible than (...), or that's what you seemed to say.

More people said that, specially with] beginning a new line. I read that people claiming square brackets as alternative to parentheses in next Scheme. I read also several people claiming square brackets for next Arc. Graham appears to like rounded ones.

Also many modern languages recently arised use [] for arrays and lists. Logo also uses []. On any case { } is characteristics of C and TeX, and () is of LISP and SXML, therefore [] appears to be a good option.

[1] <http://okmij.org/ftp/Scheme/SXML-as-database.txt>

[2] <http://my.opera.com/White%20lynx/blog/show.dml/256124>