

Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's (Vol I)

Source: <http://coding.derkeiler.com/Archive/Perl/comp.lang.perl.misc/2005-12/msg02257.html>

- *From:* robic0
 - *Date:* Thu, 29 Dec 2005 13:55:43 -0800
-

On Tue, 27 Dec 2005 09:37:12 -0500, "Matt Garrish"
<matthew.garrish@xxxxxxxxxxxx> wrote:

>
><robic0> wrote in message news:evc1r1d8jt63q25vmdocgbevd16uh5att2@xxxxxxxxxxxx
>> On Sat, 24 Dec 2005 11:57:13 -0500, "Matt Garrish"
>> <matthew.garrish@xxxxxxxxxxxx> wrote:
>>
>> Man you make me laff!
>>
>
>Well, at least you're getting as much out of this as I am. It would be nice
>if you could drop the script-kiddie talk and write proper English sentences
>in the future, though.
>
>>
>>>By the way, have you put any thought into the public interface for this
>>>thing? It's nice that it runs line-by-line and uses regexes to find tags,
>>>but that's totally useless for XML parsing. Does it handle events like a
>>>SAX
>>>parser? (Not that I see.) Does it build a parent/child tree? (Again, I
>>>don't
>>>see anywhere that you can tell what the relationship is between any set of
>>>tags.) Or is this just an exercise in writing regular expressions?
>>>
>>>
>> Since its out of sequence, its totally useless for event driven SAX.
>
>That's exactly my point. What is this thing supposed to do? The (very
>simple) point of an XML parser is to verify the integrity of the document
>(validation: either well-formedness or compliance to a dtd or schema) and/or
>allow you to access the content.
>
>Your parser has no appreciation of nesting beyond the very trivial, so there
>is no way that it can check well-formedness. It (you) also doesn't
>understand dtds or schemas, and don't realize how nearly impossible it's
>going to be for your parser to validate against one.
>

Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's (Vol I)

Whether or not I can use it to write a schema checker is something I will consider when I feel like it.

You have some misconception about the ability of schema to do 100% validation. It can't on every level, period! In fact between level sets, all it can do is validate a range of parents vs. a range of children. It can't validate a relationship between a single parent and possible several children. So schema is whoafully inadequate alone. To propagate all the possible permutations would make schema 100% valid. It doesn't have that capability and never will. If all you use is schema to validate your xml, you don't know xml..

>To get back to my original point, however, your parser does not build a >tree,

Doesn't build a tree? Wtf are you drinking?

> so that makes it useless for half the applications of a parser. It >also doesn't handle events like a SAX parser, which makes it useless for the >other half. I'm honestly curious what real world application you think this >is going to have?

To start, it blows the doors off all parsers out there. It uses a substitution method that exponentially gets quicker. It starts from the inner xml blocks and works out. It takes data off right away and builds a tree without waiting for parent closure. Its 100% accurate because the logic is flawless. It works on a micro as opposed to macro idiom. Its out of order with discrete cells. This is the fastest possible method to parse xml. I'm suprised no one ever did this before. Its on the level of a node model but it can easily hone in on patterns and discard whats not necessary and filter. All while using an examination method that exponentially gets quicker as the search progresses.

I wan't your promise that when this idea takes off that you won't have any part of it and continue to bury your head in the sand.

>
>Oh, and when are you going to start handling xpath queries?
>
>Matt
>

.

• *Follow-Ups:*

◆ ***Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's (Vol I)***

◇ *From:* John Bokma

◆ ***Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's (Vol I)***

Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's (Vol I)

◇ From: Matt Garrish

• **References:**

- ◆ **[My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: robic0
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: robic0
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: Bart Van der Donck
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: robic0
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: Matt Garrish
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: robic0
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: Matt Garrish
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: robic0
- ◆ **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
◇ From: Matt Garrish

- Prev by Date: **[Re: Modifying Array inside While statement](#)**
- Next by Date: **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
- Previous by thread: **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
- Next by thread: **[Re: My Regexp XML Parser -> Structured Perl Data, Cut & Paste Version, No Module's \(Vol I\)](#)**
- Index(es):
 - ◆ **[Date](#)**
 - ◆ **[Thread](#)**