FeedParser StringAllocationConsideredHelpful

Patrick Chanezon was nice enough to comment on our FeedFilter implementation.

His central point was that string allocation is a waste of CPU and memory and that a Reader or InputString approach would be faster/better.

I responded to the java-syndication list but figured it was better to preserve the email here as I'm sure this will come up again.

In fact I tend to link to this wiki page as documentation within the source 😉

```
patrick chanezon wrote:

>> #1 adds a memory usage penalty that can be significant in the case of
>> large feeds.
>
Actually... No it doesn't ... you already have the penalty because
you're using DOM.  If you have a 1M feed you'll need > 1M of data to
keep the DOM.

Since the string and the DOM are never in memory at the same time
there's no memory hit.

>> #3 introduces an extra char->byte conversion.
>> #4 introduces an extra byte->char conversion.
>
Worse is better ;).  Seriously... I've had to get over this a LONG time
ago.  If you think you have this problem now wait until you have to work
in an environment like Rojo. We literally are parsing strings 10-20
times and still the CPU is not our bottleneck.

It would have been nice to not to do this but our logic is too
complicated to have it in a one pass filter.

The extra CPU required is certainly worth it. I think :)

>>
>> The implementation of the HTML literal entities to coded entities
>> conversion, FeedFilter, creates a StringBuffer with a size 1000 chars
>> bigger than the String with the feed to process. Not to mention the
>> memory used by the XML stack (SAX+JDom in the current incarnation or
>> SAX in the future one, but the later we'd expect it would be not that
>> much).
>
What?  The "XML stack" and the FeedFilter aren't used at the same time
so there's no additional overhead.

>>
>> #3 and #4 could be avoided by tweaking the FeedFilter code not go
>> down to byte level, staying at String level.
>
Not really... UTF16 and UTF32 wouldn't work :-/  Latin1, UTF8, and all
the iso-* types would work but I'm worried about the other encodings
because I'm not an encoding god.

>> However, the 2x feed-size memory footprint is something that cannot
>> be removed from the current design.
>>
There isn't one ;)

>> IMO these factors can be limiting in a server environment.
>>
Not in a server environment.  So first off... there's NO overhead
because its already less than or equal to your DOM overhead which is
done *after* the parse.

Second...if this is "limiting in a server environment" then the fact
that Rome is DOM would be limiting right? ;)  This would suggest that
Rome should move to a SAX-based API ;) *cough*  ;)

So FeedParsers network API only allows files < 1M.  So if you're in a
```

HIGHLY multithreaded env with say 800 threads thats 800M of data.. Our machines are running 2G of memory and NEVER page so this has never been a problem.

I'm actually more worried about a clientside aggregator IMO due to the fact that if the user has 200 subscriptions, with threads handlign each... thats 200M of data which is TOTALLY wasted in a client aggregator.

But then again this scenario NEVER happens because the majority of feeds are < 100k.

>> The outstanding issue is then the memory utilization. I've thought
>> for a while on how this could be avoided and Algorithms 101 came to
>> the rescue, a char based stream filter.
>
but now you still have your byte -> char conversion right?

>> This is do the HTML entity conversion as the stream is read, this
>> means doing some look ahead logic.
>
which requires more memory ;)

>> This involves some extra logic for the look ahead when an entity (it
>> has been a while since I've played being lex) begins but it would be
>> comparable to what the Matcher does to find the entity patterns in
>> the FeedFilter.
>
The issue is that you're one step down the path to using real strings. We have multi-pass regexps which we use on strings mirrored from the network which would be impossible with a buffered Reader/InputStream

>>
>> I'll be checking in the code as soon as I have internet connection,
>> if you guys don't like it please do a rollback. I'd normally wait
>> until I get a few +1s but I'm going to have intermittent internet
>> access for a couple of weeks.
>>
Well you have my feedback ;)

So what does this mean exactly? You took the FeedFilter and called it XmlFilterReader (or whatever) and implemented it not using a string? I mean I'm fine with this and the license allows it but:

1. Doesn't seem like there's any benefit here
2. Prevents blackboxing the code.
3. Prevents easy cooperation on this class in the future (and stable liberal parsing between Rome and the FeedFilter)

Also if you were *right* it would have been better from a cooperation standpoint to submit a patch to the FeedFilter ... You could have just put this one class in your CVS as org.apache.commons.feedparser.FeedFilter and then we could share diffs ;)

Peace!

Kevin

--

Use Rojo (RSS/Atom aggregator). Visit http://rojo.com. Ask me for an invite! Also see irc.freenode.net #rojo if you want to chat.

Rojo is Hiring! - http://www.rojonetworks.com/JobsAtRojo.html

If you're interested in RSS, Weblogs, Social Networking, etc... then you should work for Rojo! If you recommend someone and we hire them you'll get a free iPod!

Kevin A. Burton, Location - San Francisco, CA
AIM/YIM - sfburtonator, Web - http://peerfear.org/
GPG fingerprint: 5FB2 F3E2 760E 70A8 6174 D393 E84D 8D04 99F1 4412