L2/05-121

Source: Jianping Yang (Oracle)

Date: May 3, 2005

Subject: UTS #18 Newline Handling

Our basic concern with the regular expression proposal is that it redefines the specification (POSIX, Perl, pick your flavour) of the 'any character' operator (dot) by making it match newline sequence variations as a single character.

We believe that the default behaviour (it does not normally stop at newline characters) of the 'any character' meta character should be to match any one character, defined as being a Unicode Code Point, without exceptions.

Many engines (possibly popularized from the grep utility where many of us first learned about regex) work on line boundaries so some special logic is required to stop at those line boundaries. One requirement here is that the 'any character' must not consume newline sequences and we fully recognise the requirement to handle the different newline sequence variations under these situations simply because the 'any character' meta character is being run in a 'stop at newlines' mode. It is perhaps because of the popularity of grep and perl that this association of the 'any character' and newline sequences has become so strong but often in the database newline sequences in a record are of no special meaning and merely act as a formatting character when presenting the contents. In this case we use the 'any character' meta character to match one code point at a time irrespective of the kind of code point.

Our concern is about the behaviour of the 'any character' meta character when the 'stop at newlines' mode is ***not*** enabled. In this case no special treatment of newlines should be expected as it is not defined anywhere. The 'any character' meta character knows only to match one code point at a time.

In order to come to some resolution here we propose the following ideas for discussion:

1. Enabling the 'any character' to be newline sequence aware when it is in not in a mode that requires it stops at newline sequences (what I refer to as the default mode) should be optional in the standard.

2. The document could propose some means to match 'any character' while treating newline sequences as a single character but advises against using the standard 'any character' meta character (dot) to achieve this purpose and suggests (mandates?) introducing a new meta character.

Regards,

Peter Linsley
Oracle Corporation

Makes sense as a document; I'd suggest you write it up and pass it by this group, then submit for the next UTC meeting.

Mark
___
mark.davis@us.ibm.com
IBM, MS 50-2/B11, 5600 Cottle Rd, SJ CA 95193
(408) 256-3148

Peter Linsley <peter.linsley@oracle.com>	To	Mark Davis/Cupertino/IBM@IBMUS
03.07.2005 09:59 AM	cc	Andy Heninger/San Jose/IBM@IBMUS, Jarkko Hietaniemi <jhi@iki.fi>, "Yang,Jianping" <JIANPING.YANG@oracle.com>, Weiran Zhang <weiran.zhang@oracle.com>
	Subject	Re: UTR #18

```
Indeed.

Just for the sake of continuing this interesting topic and setting a
precedence for any future discussions, I wanted to make it clear on the
way Weiran and myself see things here.

There is no problem with metacharacters that are by design supposed to
know what a line boundary looks like. Anchor and anychar stopping at
many different kinds of newline sequences when they are supposed to is
nothing but a good thing.

It is when these metacharacters are *not* in these modes that questions
begin to arise.

If we make anychar newline sequence aware at all times, it leads to
some
interesting side effects to consider for the times when anychar is in
not in the newline aware mode:

* As 1 Unicode Char no longer equals 1 Unicode Char ('.' matches 2
Unicode Chars), we can no longer handle our string as a sequence of
Unicode Characters. An expression such as .{30000} cannot zip forward
120000 bytes in a UTF-32 string but instead has to examine each and
every character to see if it's part of a DOS newline sequence. Ouch.

* Do we make everything else that processes a character 'newline
sequence' aware such as forward tracking or expressions like [^a]? If
we
don't do this there will be some inconsistency between what '.' thinks
```

is a character and what every other part of the engine thinks is a
character.

* What do '\r.' or '.\n' match? Do we promise '.' will always match
whole newline sequences or are the allowed to break into them?

Our take is that the 'anychar' metacharacter should match just that,
any
one Unicode character. This should be extended through the entire
engine
so forward tracking and so on would process on a Unicode Character
basis. This would allow people to write expressions that could
manipulate newline sequences if they wanted to do so. This
metacharacter
can then be extended to allow you to do matching on a line basis by
making it stop at a newline sequence when the appropriate mode was set.
If this can handle all flavours of newline then that's nothing but good
news for the user.

Make sense? If it would be better to write up a doc discussing the ins
and outs I'd be happy to do so. Don't hesitate to point out any errors
I
may have made in the interpretation of the various standards here.

Cheers,

Peter

Mark Davis wrote:
>
> Yes, it really sounds like "." is the only real sticking point, since
^
> and $ would be pretty simple to implement.
>
> Mark
> ___
> mark.davis@us.ibm.com
> IBM, MS 50-2/B11, 5600 Cottle Rd, SJ CA 95193
> (408) 256-3148
>
>
>
> *Andy Heninger/San Jose/IBM*
>
> 03.05.2005 11:58 AM
>
>
> To
>   Mark Davis/Cupertino/IBM@IBMUS
> cc
>   Jarkko Hietaniemi <jhi@iki.fi>, "Yang,Jianping"
> <JIANPING.YANG@oracle.com>, Peter Linsley <peter.linsley@oracle.com>,
> Weiran Zhang <weiran.zhang@oracle.com>
> Subject
>   Re: UTR #18Link
>
<Notes:///87256587001353CE/A24CF79FA927570285256356005827C4/29EF7D28A8B

```
26D9A88256FBB0000D6AD>
>
>
>
>
>
>
>
> The proposed wording is ok  by me.
>
>
> Weiran wrote
>  > I also want to point out that Java doesn't really support CRLF
> despite what javadoc states.
>
> Java does do the right thing with ^ and $, though.   ^ (start of
line)
> matches after a CR/LF.   $ matches just before one.  A pattern like
> "^.*$", run in a find() loop in multi-line mode, will correctly match
> full lines while stepping over CR/LF line endings.   You don't get a
> zero length match between the CR and the LF.
>
> -- Andy Heninger
>      heninger@us.ibm.com
>
>
>
> *Mark Davis/Cupertino/IBM*
>
> 03/04/2005 04:09 PM
>
>
> To
>                  Weiran Zhang <weiran.zhang@oracle.com>
> cc
>                  Andy Heninger/San Jose/IBM@IBMUS, Jarkko Hietaniemi
<jhi@iki.fi>,
> "Yang,Jianping" <JIANPING.YANG@oracle.com>, Peter Linsley
> <peter.linsley@oracle.com>
> Subject
>                  Re: UTR #18Link
>
<Notes:///8725658700121678/A24CF79FA927570285256356005827C4/EA5C9A7B855
2150886256FBA00812F4D>
>
>
>
>
>
>
>
> That wording works for me. Others: speak up now if you can't live
with it!
>
> Mark
> ___
> mark.davis@us.ibm.com
> IBM, MS 50-2/B11, 5600 Cottle Rd, SJ CA 95193
> (408) 256-3148
>
>
>
> *Weiran Zhang <weiran.zhang@oracle.com>*
>
> 03.04.2005 03:30 PM
>
>
> To
>                  Mark Davis/Cupertino/IBM@IBMUS
> cc
>                  Andy Heninger/San Jose/IBM@IBMUS, Jarkko Hietaniemi
<jhi@iki.fi>,
> "Yang,Jianping" <JIANPING.YANG@oracle.com>, Peter Linsley
> <peter.linsley@oracle.com>
> Subject
>                  Re: UTR #18
>
>
>
>
>
>
>
>
> We'd like to suggest the following change of wording:
>
> Note: For some implementations, there may be a performance impact in
> recognizing CRLF as a single entity, such as with an arbitrary
pattern
> character ("."). To account for that, an implementation may satisfy
R1.6
> if there is a mechanism available for converting the sequence CRLF to
a
> single line boundary character before regex processing.
>
> The main point here being the newline normalization doesn't
necessarily
> have to be part of the regex. When it's a part of regex, there should
be
> an option to switch it on or off.
>
> There are still issues relating to the exact interpretation of CRLF
as
> discussed in previous exchanges for which we will need clarification
in
> the next version. These issues are by no means special to a
particular
> implementation.
>
> I also want to point out that Java doesn't really support CRLF
despite
> what javadoc states. The attached JDK1.4 test demonstrates this. It's
> possible that they set out to support CRLF but hit similar
> implementation issues.
```

```
>
> Regards,
> -Weiran
>
> Mark Davis wrote:
>
>  >but we haven't been able to reach consensus on this.
>
> I don't know who the "we" was. I have been trying to be accommodating
> here, but this could have been discussed a *long* time ago. I sent a
> message on 11.18.2004 09:40 AM to Peter Linsley including the
following:
> ...
> 5. The remaining issue was the specification of ".", when handling
line
> separators. What we agreed to do was to add some explanatory text
about
> the issue, so I am sending out this message to you all so that we can
> come up with and agree on some language to add. Please look at the
> message from Weiran Zhang below.
> ...
>
> That indicated the consensus that had been arrived at during the UTC
> meeting in November. I didn't hear anything back after that memo
until
> *after* the UTC meeting in early February, where it was too late to
make
> substantial changes.
>
> Anyway, I don't want to rake over old coals -- let's just see about
> getting some language that can be worked with. You said:
>
> Note: For some implementations, there may be a performance impact in
> recognizing CRLF as a single entity, such as with an arbitrary
pattern
> character ("."). To account for that, an implementation may satisfy
R1.6
> if it provides a mechanism for converting the sequence CRLF to a
> single line boundary character before regex processing.
>
> I think we are moving in the right direction but there are some
problems
> with this proposal. First of all, it says "an implementation may
satisfy
> R1.6 if it provides a mechansim for converting the sequence CRLF
...",
> which indicates the newline normalization should be part of the regex
> implementation. The normalization cost can be expensive for large
size
> data and adds overhead to regex processing so the performance issue
is
> still there. Secondly, the normalization process may alter the input
and
> impact the match result. Suppose a pattern specifically looks for the
> CRLF sequence which exists in the original input, it will no longer
be
> able to match after the normalization. In such cases, normalizations
```

```
are
> not acceptable since it changes the match result.
>
> Comments:
> 1. It actually doesn't require much at all. It doesn't say even that
> newline normalization is part of the regex, just that it has to be
> available. And any regex I know of allows the replacement of all CRLF
> pairs by a single, say, LF.
> 2. Notice that it doesn't say that anyone has to do this replacement,
> just that it has to be available.
> 3. If the "normalization" didn't alter the match result, it would be
> pretty pointless, since the whole goal is to alter the match result
so
> that "." matches line separators including what was CRLF.
> 4. This does not attempt to say that it is a complete solution; it is
> just to say that someone could work around the issue by appropriately
> changing the patterns.
>
> Now, I'm not stuck on this wording; if you want to suggest other
> wording, please do.
>
>
> Mark
> ____
> __mark.davis@us.ibm.com_ <mailto:mark.davis@us.ibm.com>
> IBM, MS 50-2/B11, 5600 Cottle Rd, SJ CA 95193
> (408) 256-3148
>
>
> *Weiran Zhang **_<weiran.zhang@oracle.com>_*
> <mailto:weiran.zhang@oracle.com>
>
> 03.03.2005 04:43 PM
>
>
> To
>                  Mark Davis/Cupertino/IBM@IBMUS
> cc
>                  Andy Heninger/San Jose/IBM@IBMUS, Jarkko Hietaniemi
_<jhi@iki.fi>_
> <mailto:jhi@iki.fi>, "Yang,Jianping" _<JIANPING.YANG@oracle.com>_
> <mailto:JIANPING.YANG@oracle.com>, Peter Linsley
> _<peter.linsley@oracle.com>_ <mailto:peter.linsley@oracle.com>
> Subject
>                  Re: UTR #18
>
>
>
>
>
>
>
>
>
>
> Well, we did raise the newline issue back in last October when UTC
was
```

```
> seeking public review comments on UTS#18, but we haven't been able to
> reach consensus on this.
>
> 1. Adding a note in the new version to the effect that:
>
> Note: For some implementations, there may be a performance impact in
> recognizing CRLF as a single entity, such as with an arbitrary
pattern
> character ("."). To account for that, an implementation may satisfy
R1.6
> if it provides a mechanism for converting the sequence CRLF to a
> single line boundary character before regex processing.
> I think we are moving in the right direction but there are some
problems
> with this proposal. First of all, it says "an implementation may
satisfy
> R1.6 if it provides a mechansim for converting the sequence CRLF
...",
> which indicates the newline normalization should be part of the regex
> implementation. The normalization cost can be expensive for large
size
> data and adds overhead to regex processing so the performance issue
is
> still there. Secondly, the normalization process may alter the input
and
> impact the match result. Suppose a pattern specifically looks for the
> CRLF sequence which exists in the original input, it will no longer
be
> able to match after the normalization. In such cases, normalizations
are
> not acceptable since it changes the match result.
>
> Thanks,
> -Weiran
>
>
> Mark Davis wrote:
>
> I ran this by a few other members, and I think we will get a lot of
> resistance to the ballot, especially since this could have been
brought
> up long before now.
>
> However, I did get the editorial committee to agree on the following.
>
> 1. Adding a note in the new version to the effect that:
>
> Note: For some implementations, there may be a performance impact in
> recognizing CRLF as a single entity, such as with an arbitrary
pattern
> character ("."). To account for that, an implementation may satisfy
R1.6
> if it provides a mechanism for converting the sequence CRLF to a
> single line boundary character before regex processing.
>
> 2. In the *next* version of UTS #18, adding an explicit R2.7 clause
for
```

```
> those that want to indicate that they don't need the above note (text
to
> be worked out later).
>
> I think this probably satisfies Oracle and Perl's requirements.
Please
> get back to me on this asap.
>
>
> Mark
> ____
> __mark.davis@us.ibm.com_ <mailto:mark.davis@us.ibm.com>
> IBM, MS 50-2/B11, 5600 Cottle Rd, SJ CA 95193
> (408) 256-3148
>
> *Weiran Zhang **_<weiran.zhang@oracle.com>_*
> <mailto:weiran.zhang@oracle.com>
>
> 03.03.2005 12:42 AM
>
>
> To
>                   Mark Davis/Cupertino/IBM@IBMUS
> cc
>                   Andy Heninger/San Jose/IBM@IBMUS, Jarkko Hietaniemi
_<jhi@iki.fi>_
> <mailto:jhi@iki.fi>, "Yang,Jianping" _<JIANPING.YANG@oracle.com>_
> <mailto:JIANPING.YANG@oracle.com>, Peter Linsley
> _<peter.linsley@oracle.com>_ <mailto:peter.linsley@oracle.com>
> Subject
>                   Re: UTR #18
>
>
>
>
>
>
>
>
>
>
>
>
> It looks good to me. Thanks.
>
> Regards,
> -Weiran
>
> Mark Davis wrote:
>
> It sounds like that is the consensus. Here is a rough draft of the
> changes we would suggest something like the following:
>
> UTS #18 Version 10 has been approved by the UTC for release. However,
> after the UTC meeting, there was further discussion on the line
boundary
> issue raised by Oracle. After considering the line boundary issue at
> some length, we call for a letter ballot on the following change to
UTS
> #18, to make recognition of CLRF as a single entity be a requirement
of
> Level 2, not level 1. The goal is to have the letter ballot complete
as
> soon as possible, so that release of UTS #18 is not delayed much (at
> all) after the release of Unicode 4.1, since the property value
> recommendations are closely related to that release.
>
> The changes would be:
>
> In 1.6 (_http://www.unicode.org/reports/tr18/#Line_Boundaries_)
> a) Remove CLRF from the requirements for R1.6 and in the text
> b) Add a note:
> For some common implementations, there may be a performance impact in
> recognizing CLRF as a single entity, such as with an arbitrary
pattern
> character ("."). Thus this feature was moved from Level 1 in previous
> versions of this document to Level 2. See Section 2.7 Multicharacter
> Line Boundaries
>
> Add a new section 2.7 Multicharacter Line Boundaries
> a) Add a requirement R2.7 which adds CLRF to the list in 1.6
> b) Indicate the addition of CLRF to *Logical line number* and
*Arbitrary
> character pattern (often ".")* and the "\R" discussion.
> [I don't think we need it in the others]
>
>
> Mark
> ____
> __mark.davis@us.ibm.com_ <mailto:mark.davis@us.ibm.com>
> IBM, MS 50-2/B11, 5600 Cottle Rd, SJ CA 95193
> (408) 256-3148
> *Weiran Zhang **_<weiran.zhang@oracle.com>_*
> <mailto:weiran.zhang@oracle.com>
>
> 03.02.2005 05:14 PM
>
>
> To
>                  Mark Davis/Cupertino/IBM@IBMUS
> cc
>                  Jarkko Hietaniemi _<jhi@iki.fi>_
<mailto:jhi@iki.fi>, Andy Heninger/San
> Jose/IBM@IBMUS, "Yang,Jianping" _<JIANPING.YANG@oracle.com>_
> <mailto:JIANPING.YANG@oracle.com>, Peter Linsley
> _<peter.linsley@oracle.com>_ <mailto:peter.linsley@oracle.com>
> Subject
>                  Re: UTR #18
>
>
>
>
>
>
```

```
>
>
>
>
>
>
>
>
> Hi Mark,
>
> Just checking to see if there's any update on this. I think this
issue
> is worth doing the ballot for.
>
> Thanks,
> -Weiran
>
> Weiran Zhang wrote:
> That's exactly the kind of ambiguities we were concerned about. Even
if
> we work out a complete specification of the intended semantics, it
will
> entail additional string processing complexity to handle these
sequences
> uniformly across all constructs, which makes it undesirable to be
part
> of the default behavior.
>
> Thanks,
> -Weiran
>
> Jarkko Hietaniemi wrote:
> We also need to clarify exactly what the CR/LF requirement means.
 Does it
> only impact the determination of line beginnings and endings for ^
and $,
> and what is matched by "." in DOT ALL mode, or does it also mean, for
> example,  that \u000a compares equal to \u2028 in any context?
>
>
> This is true.  Regardless of what definition we end up with, just
saying
> "CR LF should match like LF" is not enough.  As Andy says, the exact
> definitions of ^, $, ., and what about negation (of \n) in character
> classes/properties, and what is the induced difference, if any,
between
> single-line and multiline matching, all these definitions need to be
> inspected.
>
>
>
> import java.util.regex.Pattern;
> import java.util.regex.Matcher;
>
> public class RXTest
> {
>  public static void main( String[] args )
>  {
>    String nl_seq = new String("\r\n");
>    Pattern p = Pattern.compile(".", Pattern.DOTALL);
>    Matcher m = p.matcher(nl_seq);
>    boolean b = m.matches();
>    if(b)
>      System.out.println("Matched CRLF");
>    else
>      System.out.println("Did not match CRLF");
>  }
> }
>
>
>
```