Conversion Tools - Bug #2262

ON statement preprocessing and parsing issues related to unmatched quotes or tilde

03/19/2014 11:13 AM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 03/19/2014 12:26 PM - Greg Shah

This testcase (checked in as on_stmt_event_as_single_or_double_quote.p) shows the issues:

```
def var txt as char.

define frame f0 txt.

on ~" anywhere
do:
   message "double quote".
end.

on ~' of txt
do:
   message "single quote".
end.

on ~~ of txt
do:
   message "tilde".
end.

enable all with frame f0.
wait-for go of frame f0.
```

There are multiple problems here:

1. The preprocessor will fail badly when it hits a ~" or ~' (unmatched quote chars). Basically, this case is treated as an opening quote for a string and then the rest of the file will be wrong because the string parsing will absorb content it is not supposed to (up to next opening quote of a matching type). This should be reasonably straightforward because we just have to avoid string parsing mode when there is a leading tilde.

2. The parser will fail (in ON statement parsing) for the same reason. This one is going to be very hard because the tildes are removed. I do think that the string parsing for an ON stmt event can tell the difference between an unmatched quote and a string like "some thing". Interestingly, when you embed whitespace in a string, the entire contents are tested to see if it is a valid event and you get an error (because no events exist with embedded whitespace, as far as I know). Another interesting finding is that ~"choose~" is reported as NOT a valid event but "choose" is accepted without problem. So the 4GL has syntax parsing with some knowledge that we don't currently provide. Part of our problem in solving this is the lexer side where the STRING matching occurs is disconnected and often driven by lookahead. This means changing the lexer behavior based on context is very hard (sometimes impossible). Even if the preprocessor stores hints about the leading tilde, there will probably be issues with content-specific lexing.

3. The TILDE is skipped/hidden right now and it should not always be. It turns out that there are cases where the TILDE is visible in the code and cannot be inserted. I believe that only tildes that are at the end of a line will be ignored. If there is (non-whitespace) text after a tilde, the tilde is not removed. This will require changes to the lexer and TILDE probably needs to be put into the UNKNOWN_TOKEN rule so that it can be matched in the ON stmt as an event.

Manual edits can easily make this same testcase preprocess/parse properly (enclose all 3 event specifications in a real string):

```
def var txt as char.
```

```
define frame f0 txt.

on "~"" anywhere
do:
   message "double quote".
end.

on "~'" of txt
do:
   message "single quote".
end.

on "~~" of txt
do:
   message "tilde".
end.

enable all with frame f0.
wait-for go of frame f0.
```