FIRST - Bug #4005

Question separator function logic fails while parsing Exam2.pdf

04/21/2009 09:59 AM - Sandeep Namilikonda

Status:	Resolved	**Start date:**	04/21/2009
Priority:	Normal	**Due date:**	
Assignee:	Sandeep Namilikonda	**% Done:**	0%
Category:	client	**Estimated time:**	0.00 hour
Target version:	post-clientprototype	**Spent time:**	0.00 hour
Bugzilla-Id:	4005		

Description

While parsing the following question, the current logic deems the text "12 S" of choice B, second line, as a part of an Answer Key and hence, deletes all the subsequent questions!

37) Some bacteria can undertake a type of
photosynthesis that uses H2S in place of
H2O. Assuming that the process is
otherwise similar to green plant
photosynthesis, which of the following
could represent the overall reaction?
A) 6 CO2 + 12 H2S > C6H12S6 + 6 H2S + 6
O2
B) 6 C02 + 12 H2S > C6H12O6 + 6 H2O +
12 S
C) 6 CO2 + 12 H2S > C6H12O6 + 6 H2S +
6 O2
D) 6 CO2 + 12 H2S > C6H12S6 + 6 H20 + 6
S02

Here is an excerpt from the Debug Console:

```
Question Num Read 36
%%%%%%%%FOUND NUMBER IN SEQUENCE%%%%%%%%%%%%
     36
    Question Num Read 37
%%%%%%%%FOUND NUMBER IN SEQUENCE%%%%%%%%%%%%
     37
    Question Num Read 12
+++++++ADDED BECAUSE OF NEW PAGE+++++++++
Block deleted: #41
Block deleted: #41
Block deleted: #41
Block deleted: #41
```

and so on!

Later, when the user tries to save the assessment, a NullPointerException is thrown in recognizeQuestions() function.

History

#1 - 04/21/2009 10:36 PM - Sandeep Namilikonda

Fall05-Exam3.pdf:

```
- Parsing stops after Q36 while reading the answer key on page 7.
      The actual document has 13 pages with Q37-51 on later pages.
```

Looks like the logic in the PDFAssessment deems answer choice content as AnswerKey! and deletes all the succceeding questions

```
if(inAnswerKey){
                //once in answer key, remove all leftover nodes.
                System.err.println("Block deleted: #"+i);
                doc.remove(i);
                i--;
```

```
}
```

#2 - 04/22/2009 05:40 AM - Ryan McFall

Yes, I noticed this too; this is what I was looking for yesterday when I sent Sandeep the email.

Right now it assumes that if it sees a number at the beginning of a line that doesn't match the expected number it automatically goes into answer key mode.

This seems like a terrible assumption. At the very least, it seems like we should scan down the subsequent lines and see if we find the next expected number later on in the document, and if so, assume that this line is part of the current question.

Also, doesn't it seem to make sense that the answer key (if it exists) would start with the number of the first question?

The answer key seems like a rare enough thing that perhaps we shouldn't even try to handle it - wouldn't ignoring pages be the right way to go?

I'll handle this one, because I want to clean this code up anyway. It's quite ugly, wouldn't you agree?

#3 - 04/22/2009 06:05 AM - Sandeep Namilikonda

Yes. I was thinking about re-writing the code too. But, if you want to do the clean-up that is fine by me although I would not mind working on one of the main functions in the overall question recognition process. One of the things that will greatly help us is if we had a pseudo code/flow chart for the various conditions that need to be checked to satisfactorily handle the various cases.
Is that something you think I should invest some time building?

#4 - 05/04/2009 02:08 PM - Sandeep Namilikonda

Faulty AnswerKey detection logic resulted in this erroneous behavior, which has been rectified now.

The solution has been tested on Exam2.pdf and Fall05-Exam3.pdf (latter actually has answer key text in it).

#5 - 05/11/2009 07:32 AM - Ryan McFall

(Resubmit after fire)

In trying to handle the answer key, I see you're showing an error message in
PDFAssessment2_4.java and then calling System.exit () inside of the
separateQuestions method.

We definitely don't want to call System.exit; this will cause the whole Morpho
application to exit, and any metadata that has been entered may be deleted.

I think it would be better for us to define an Exception class that represents
a recognition exception. Then the code that is using PDFAssessment2_4 could
provide the user with a way to fix the error rather than making them start over
from scratch. This would involve showing the wizard page that specifies
regions to be ignored outside the context of the wizard. The class that does
this is SelectIndividualText in the package edu.msu.first.parser.gui.wizard.
It will require that you populate a WizardData object with the appropriate
information (it would be the information saved in AssessmentFileSelection, also
in that package).

Can you give this a shot? I can help if you need more information.

#6 - 05/11/2009 06:08 PM - Sandeep Namilikonda

AnswerKey recognition happens in separateQuestions(). Each line of text is dealt with separately in this function. So, when an answerkey is detected, this means either the detection of string "answerKey" or "<number>) <letter>"!
If the user is presented a wizard to choose text to ignore, we have to do it relative to the beginning of the current page, in which case, we have to precisely know the index of the first valid line of text on the current page.
This way, the text prior to the current page doesn't need to be re-processed.
This solution can be applied to any other recognition error and point out the text that resulted in the error. In fact, the AnswerKey issue can be resolved very simply by just disposing the lines of text that match the two cases detailed above.

Ryan, could you please comment on this?

#7 - 05/12/2009 07:42 AM - Ryan McFall

I'm not sure why you are saying that we need to know the index of the first valid line of text on the current page. We would know where we ~~think~~ the answer key starts (although I don't remember if StringBlocks contain geometry information). We could then mark up the PDF with a suggestion for the answer key by selecting the entire portion of the document from there on if we wanted to.

I don't think the answer key problem can be solved by getting rid of lines that look like the ones that separateQuestions uses to guess about being in the answer key. The subsequent lines after the beginning of the answer key need to be ignored as well.

I'd still like to see some more intelligent guessing about whether we're in the answer key or not. The presence of the words Answer Key is obviously a pretty good indication. However, the <number>) <letter> thing seems fairly arbitrary to me. It works for multiple choice questions, but not any other type. To me, it seems likely that we are in the answer key if the following conditions hold:
- The out of order number that we think is the beginning of the answer key is the same number as the number for the first question
- We see numbers after the first out of order number for all questions on the assessment
- There aren't any numbers larger than the last one encountered before the answer key started after the answer key

```
What do you think?
```

#8 - 05/13/2009 12:28 PM - Sandeep Namilikonda

The logic you (Ryan) presented is more sound. Clearly, the current logic to recognize Answer Key cannot handle any question type except multiple choice questions. But, checking for out of order numbers can also be a potential solution for detecting numbered paragraphs such as code snippets, pseudo code, or steps of an experimental procedure that form a part of a question. Hence, displaying the wizard to the user as you suggested with an option to discard a portion of text or to include it as a part of an existing question (in case, the text block is a code snippet, for instance) is a better solution.

The reason I thought we need to know the index of the first valid line of text on the current page is to track the location of that block relative to the beginning of the page. But, each "StringBlock"'s attributes already have that information! My bad!

#9 - 05/14/2009 06:26 PM - Sandeep Namilikonda

We have reached a consensus that this case will not be thoroughly handled for the client-prototype release. We will go ahead with the assumption that the Answer Key, if present in an exam, will appear at the end of the document, which may result in the answer key text being clubbed with the last recognized question. It is assumed that the user will then manually edit that question or prune the irrelevant text from the original document and re-process it. Of course, a user can always use the "ignore" tools available in the assessment importer wizard to discard irrelevant text such as the answer key.

#10 - 03/27/2013 02:25 PM - Redmine Admin

Original Bugzilla ID was 4005