Auto Multiple Choice - Feature # 91: count questions and (max) points per category

Status:	New	**Priority:**	Normal
Author:	Pieter Van den Hombergh	**Category:**	
Created:	11/06/2012	**Assignee:**	
Updated:	03/17/2024	**Due date:**	
Description:	It would be help full to be able to pick up the total number of questions in an exam as well as the maximum points per category. Our exam practice often has two question categories, _general_, mc questions, to be shuffled each question with typically the same max number of points per question and _open_ questions, not shuffled and typically a different number of points. It would be nice to be able to count the questions and the maximum number of points per category to be able to use this information to be put on the cover page of the exam. Our regulations stipulates that we put this information on our cover page. Probably these counters are already implemented but not documented for public use.		

History

11/06/2012 02:27 pm - Alexis Bienvenüe

AMC processes the scoring strategy and all scoring computations within the perl programs, not with LaTeX. LaTeX is only used to pass to the perl program the scoring strategies and the wrong/right answers. So I'm afraid the only solution is to hard-code the maximum number of points in the LaTeX source, and maintain the correspondence between this value and the scoring strategy...

11/06/2012 06:02 pm - Pieter Van den Hombergh

- File countpoints.pl added

Then probably some perling on my side can also help.
Since I can either infer from the questions how many points they get or add an annotation per question file. (As some kind of structured comment as in %%maxpoints=3)

Since we put the reference to questions in separate file per category, like
<pre>
\element{general}{\inputQ{02threadsafety/oProperGuarding.tex}}
\element{general}{\inputQ{02threadsafety/oThreadSafeCode1.tex}}
</pre>
summing the points should be easy enough.

Is it possible to add a pre-update hook in the process cycle?
This pre-update hook would be a command/script, specified either in the global settings or per project.
It should be invoke just before the latex processor (like pdflatex). This is what I would do if I would use a make file to compile the text.

This could then easily run such a script as mentioned above before the latex run.
Such script would produce a file with a _well known name_, such as counts.tex, which then again could be inputted (or \InputIfFileExists{}) in the preamble, et voilà, we have what we need.

Find simple script attached. It is just a hack, fitting to our practice.

11/12/2012 09:25 pm - Alexis Bienvenüe

- Category deleted (LaTeX)

The problem is difficult to solve for all exams, because the scoring strategies are allowed to be quite complex, and the questions given to different students may vary, as well as the maximum scores for a given question...
I can see two possible solutions for your problem, although I must admit none of these are very good.

1. Using LaTeX

Define yourself the counter to be initialized at the beginning of a copy, and add the max scores for each question in turn. You may define some LaTeX macros to get a simpler LaTeX code...

2. Using perl

If you know how to process the source file from a perl program to get what you need, perhaps you can declare a new "source file format" as a plugin - see a very quick howto at [[Build a new filter plugin]].

11/12/2012 10:44 pm - Pieter Van den Hombergh

Alexis Bienvenüe wrote:
> The problem is difficult to solve for all exams, because the scoring strategies are allowed to be quite complex, and the questions given to different students may vary, as well as the maximum scores for a given question...
> I can see two possible solutions for your problem, although I must admit none of these are very good.
>
> *1. Using LaTeX*
>
> Define yourself the counter to be initialized at the beginning of a copy, and add the max scores for each question in turn. You may define some LaTeX macros to get a simpler LaTeX code...
>
> *2. Using perl*
>
> If you know how to process the source file from a perl program to get what you need, perhaps you can declare a new "source file format" as a plugin - see a very quick howto at [[Build a new filter plugin]].

Your proposal would lead some kind of generic solution, which might be quite complex. Actually, my requirements are rather more humble. I would like to avoid to have to specify the question count and the
maximum total score, when they should theoretically be computable from the input. If LaTeX does not maintain such numbers during a run, then summation of the score and counting the questions could be done in
an external program, e.g. in perl.

Our way of creating exams uses a fixed set of files, in a directory specific for one exam (very much as AMC does itself)
The files are called *source.tex* and *questions.tex* for mc-only exams. The source file defines so meta data on the exam such as date and time.

The questions.tex file only contains lines as above \element{general}{\inputQ{02threadsafety/oThreadSafeCode1.tex}}, which is easily
parsed and processed by the custom perl script that knows of these files and also can derive where the actual question files can be found, in this case for instance, where the subdir 02threadsafety can be found.

I really have no reason to define another file type, I am quite glad with LaTeX.

What would help is somehow run the mentioned perl script as part of the latex processing, so you will not forget. Now I have to run the script by hand every time I add or change a question file. This is not a big deal, but running it from the AMC process would be simple. I trick came up my selves is make a pdflatex wrapper, an executable script that first calls the perl-script and after that the normal pdflatex.

11/13/2012 11:50 am - Alexis Bienvenüe

> I really have no reason to define another file type, I am quite glad with LaTeX.

The file type I proposed to you was a _virtual_ file type, like "LaTeX with a small preprocessing"...

I added a preprocessing option in LaTeX source files from revision r1216: add the line
<pre>
%%AMC:preprocess_command=countpoints.pl
</pre>
to your LaTeX source (with no non-comments lines before this one). This will tell AMC to call
<pre>
countpoints.pl DOC-filtered.tex
</pre>
from the project directory, after having copied @source.tex@ to @DOC-filtered.tex@ (so that your command may change the contents of @DOC-filtered.tex@ with no problem).
You can write the full path of the command if it is not in the project directory.

03/17/2024 11:19 pm - Michaël Cadilhac

FYI, I did something similar, but in LuaTeX. I have a small Lua script that parses the .amc file at '\end{document}' and produces a list of scores and some more info. This is then put in the .aux file, and when reran, the document has access to all the info.

Files

countpoints.pl	1.3 kB	11/06/2012	Pieter Van den Hombergh