AcroTEX.Net

The pmdb Package

D. P. Story

Prepared: June 7, 2021

www.acrotex.net
Version 1.0, 2021/06/07

Table of Contents

1 Introduction 3
2 Requirements and options 3
3 The DB stage 4
4 The production stage 6
5 Package commands 6
6 Comments on portability 9
7 Final comments 9

1. Introduction

This package addresses the issue of a poor-man's database (pmdb).[1] Educators who use LaTeX to construct exams and homework sometimes have a collection of problems. Each problem is in its own TEX file. When the educator creates a new exam or homework document, a "common" workflow is to \input several of these prepared questions. This package attempts to provide a visual "user interface" to the questions and to provide a mechanism for viewing and selecting questions that are to be included in the document.

\pmInput{⟨path⟩}

How does this package operate? For a document that inputs content using the LaTeX command \input, the same content can be input using the command \pmInput, a command defined in this package. When content is input by \pmInput, a checkbox is created in the margin at the insertion point of the content. The checkboxes so created can be checked (to select the associated content) or cleared (to de-select the content). When the user clicks on a push button provided by this package, a list of all *selected* \input statements is displayed in the JavaScript console. This list can then be copied and pasted into another document the author is developing. If you Ctrl+Click on a checkbox, the associated content is opened in the default browser. For this workflow, the document author can see a typeset of the content and decide whether the content should be included in the developing document, and can optionally view the source file to edit it by clicking on the button or link to the right of the checkbox.

Ctrl+Click

2. Requirements and options

aeb-reader.js

Folder JavaScript. The 'Ctrl+Click' action feature requires the installation of the folder JavaScript file aeb-reader.js,[2] found in the folder-js folder of this distribution. This file comes with the distribution of the pmdb package. If you already have the aeb_pro package, you've already installed the file aeb_pro.js, which includes the special JavaScript functions use by pmbd; however, version 1.7 or later of aeb_pro.js is required. Download the latest version of aeb_pro if Version 1.7 is not on your system already. The installation procedure of folder JavaScript files is described in the file docs/install_jsfiles.pdf. The folder JS file aeb-reader.js (or aeb_pro.js) enables the 'Ctrl+Click' to be operational for both Adobe Acrobat (AA) and Adobe Acrobat Reader (AR). However, when the command \editSourceOn is expanded, a pushbutton appears to the right of the checkbox; click on this pushbutton opens the content in the default editor. The use of the button generated by \editSourceOn requires neither aeb-reader.js nor aeb_pro.js.

Version 1.7 required

AA and AR

\editSourceOn

dbmode
!dbmode
tight
!tight

Options. There are four options: dbmode and !dbmode, and tight and !tight.[3] The default options are dbmode and !tight. When dbmode is in effect, checkboxes appear in the margins at each \pmInput point; for the option !dbmode, the checkboxes are not produced. The use of an exclamation point (!) makes it convenient to turn on or off the creation of the marginal checkboxes. The default location of the checkboxes are flush

[1] The basic concept for this package was suggested to my by Thorsten G.

[2] If you don't see a need for this feature, the installation of aeb-reader.js is not essential.

[3] These options set Boolean switches, \ifpmdbmode and \ifpmdbtight, the state of these switches can be changed within the body of the document.

left. The `tight` option places the checkboxes "tight" up against the text area. Refer to the **Marginal Checkboxes** paragraph for more information.

Requirements. The `eforms` package is required for the creation of checkboxes and pushbuttons.

3. The DB stage

When you have a collection of questions (or content) in various files, this package enables you to build a document that displays these questions (or content) in a single 'DB' document. Once your DB document is build, you can use the checkboxes in the margin to select content you want to include in another document; you can use the Ctrl+Click feature to view the source file of that content as well.

The following comments are apropos to the creation of a DB document:

- **PDF creators:** Any PDF creator current in the LaTeX world is valid for use with this package.
- **PDF viewers:** The ideal viewer is AA; however, AR and PDF-XChange Editor can also be used. In the case of Adobe Reader, there is an annoying security dialog box that appears each time you use the Ctrl+Click feature of the checkbox;[4] the Ctrl+Click feature *does not work* with PDF-XChange Editor.

To remove the security warning

For AR, the annoying security dialog mentioned above is emitted when AR is in Protected View. To avoid the security dialog, exit Protected View as follows: (1) open Preferences (Ctrl+K) of AR; (2) select Security (Enhanced) from the left panel; and (3) clear the Enable Protected Mode at startup checkbox. For AA, Protected View set to Off by default.

Outline of a DB file. A DB file is just a LaTeX document that uses the pmdb package. The document itself uses `\pmInput` to input its content.

```
\documentclass{article}
\usepackage[forcolorpaper]{web} % optional
%  Additional packages that may be required by any content
%  that is input with \pmInput. For example, ...
\usepackage{exerquiz} % if needed
\usepackage[dbmode]{pmdb}
...
\editSourceOn % or, \editSourceOff
%\useEditLnk  % \useEditBtn, the default
...
\begin{document}

% Declares input for quiz items
%\InputQuizItems
% Declares input paragraph content
%\InputParas
```

[4] This assumes the file `aeb-reader.js` is properly installed.

```
% Declares input items
%\InputItems

\pmInput{⟨path1⟩}

\pmInput{⟨path2⟩}

...

\pmInput{⟨pathn⟩}

\displayChoices\quad\clrChoices
\end{document}
```

Descriptions of the various commands \pmInput, \InputQuizItems, \InputParas, \InputItems, \InputProbs, \editSourceOn, \displayChocies, and \clrChoices appear later in this documentation.

These methods are not restricted to inputting quiz items or whole chapters. This paragraph was input into the main document with \pmInput{sample-para.tex}. Note the check box in the left margin. If you have aeb-reader.js or aeb_pro.js properly installed, you can Ctrl+Click to see this paragraph at the source.

Functionality of the boxes in the margin. By default, only one box appears in the margins, a checkbox. If the command \editSourceOn is in effect, a small pushbutton appears to the left of the checkbox.

checkbox: Selecting the checkbox (a check mark) appears declares that you want that problem (or item) in the document you are creating. Ctrl+Click opens the source file for viewing (not editing) in the default browser. A Shift+Click action jumps—if \InputQuizItems is in effect—to the solution of the selected item, if a solution is provided. The Ctrl+Click functionality requires the successful installation of the aeb-reader.js or aeb_pro.js JavaScript file.

pushbutton: If \editSourceOn has been expanded prior, a little pushbutton appears to the right of the checkbox. Clicking the pushbutton opens the default viewer (for a TEX file) and the source file is loaded into the viewer for possible editing.

link annotation: If \editSourceOn and \useEditLnk are expanded, a link annotation having the same functionality as the pushbutton appears in the margins, in place of the pushbutton.

The checkbox/pushbutton pair above have been disabled for this documentation. Experience the functionality with the example files, listed below.

Sample files. The four sample files are found in the examples folder:

- tst-qzdb.tex: The example that motivated the creation of this package. Input various quiz questions into an quiz environment of exerquiz.
- tst-paras.tex: For the book class, we input chapters of the book using the command \pmInput.

- `tst-qzdb-paras.tex`: A combination of the two example files above, were we declare `\InputQuizItems` to input content of a quiz, and `\InputParas` to input chapters.
- `tst-items`: An example that demonstrates the `\InputItems` input mode.
- `tst-eqedb.tex`: An example that demonstrates the `\InputProbs` input mode.

4. The production stage

After your DB document has been assembled (using pmdb), you are ready to use your DB document to select questions (for quizzes) or other content for insertion into a new document.

Open your DB document in AA or AR, and open the source file of your developing document in your LaTeX editor. Within the DB document, select questions or content by checking any of the checkboxes in the margin. Now press the `\displayChoices` push button. The AA (AR) console window opens and displays your choices; for example,

```
\input{probs/prob1.tex}
\input{probs/prob4.tex}
\input{probs/prob5.tex}
```

These can be copied and pasted into your document. Use the Ctrl+Click feature to view the sources of your choices. It may be you want to modify the source for your document; (1) edit the DB source snippet you are inputting; or (2) copy and paste the whole content into your document and make the needed changes there. Once all content has been referenced by your developing document, you can compile into a PDF. Done!

5. Package commands

The package defines several commands and these are discussed now.

`\pmInput*[⟨arg⟩]{⟨path⟩}`

Within a DB source document, content is inserted using `\pmInput`. This command both inputs the referenced ⟨*path*⟩ (using the LaTeX command `\input`) and places a checkbox in the margin. The ⟨*path*⟩ can be a relative or full path reference. If the ⟨*path*⟩ contains any spaces, the path needs to be enclosed in *double quotes* ("); for example,

spaces in ⟨path⟩ discussed

```
\pmInput{"C:/Users/Public/Documents/My TeX Files/tex/%
  latex/aeb/pmdb/examples/chapters/doc2.tex"}
```

The optional argument ⟨*arg*⟩ is only obeyed when `\InputItems` is active and `\pmInput` is expanded within a list environment; ⟨*arg*⟩ is passed to the underlying `\item` in the list (`\item[`⟨*arg*⟩`]`). When the * option is taken, the rest of the arguments are gobbled and the command does nothing; this is a convenient way of *not inputting* a ⟨*path*⟩.

extension required

Important requirement: Unlike the normal `\input` command, we require the file name to include the extension, '`.tex`' in the above example.

Input modes. There are four 'input modes':

`\InputParas`	(the default)
`\InputQuizItems`	
`\InputItems`	
`\InputProbs`	

A brief description of each follows.

`\InputParas`: Sets the input mode to input 'paragraph content'. This input mode is suitable for exercises created by the `exercise` environment of `exerquiz`, whole paragraphs, or whole chapters.

`\InputQuizItems`: Sets the input mode to input items in a quiz (as created by the `quiz` environment).

`\InputItems`: Sets the input mode to input items in a list environment.

`\InputProbs`: Set the input mode to input problems for an exam created by the `eqexam` package.

`\marginpar` used

In all cases, the LaTeX command `\marginpar` is used; as a result, the checkbox appears in the margins when the `\marginpar` command is supported; in particular, `\marginpar` does not work in a `tabular` environment or a `multicols` environment, for example.

Below is an example of `\InputItems`.[5]

- This is content destined for an list environment and was input by `\pmInput`.
- Another item, not input by `\pmInput`

E * This is content destined for an list environment and was input by `\pmInput`.

The verbatim listing is,

```
\begin{itemize}\pmdbtighttrue\InputItems
\pmInput{sample-item.tex}
\item Another item, not input by \verb|\pmInput|.
\useEditLnk % Use marginal link for illustrative purposes
\pmInput[*]{sample-item.tex}
\end{itemize}
```

Note the use of the optional argument for the last `\pmInput`. Note also that both `\pmdbtighttrue` and `\InputItems` are expanded locally. When `\pmdbtighttrue`, the checkboxes appear "tight" against the text box margin (flush right, in this case). You can (locally) move the checkboxes to the right margin by expanding `\normalmarginpar` within the `itemize` environment group.

[5] The marginal form fields have been made readonly for this documentation.

Marginal Checkboxes. The document produces checkboxes in the margins, you can set the appearance of the checkboxes using the `\pmCBPresets` command.

```
\pmCBPresets{⟨opts⟩}
```

Pass eforms key-value pairs to the checkboxes through the argument ⟨*opts*⟩; for example, this document was compiled with `\pmCBPresets{\textColor{red}}` declared in the preamble, as a result, the checks are colored red.

The checkboxes are placed in the margins and hopefully correctly aligned at the insertion point. For `article` class-type documents, use of `\reversemarginpar` is recommended. In this case, the checkboxes appear in the left margin at the extreme left (as seen above). For `book` class-type document, the checkboxes alternate between the left and right margins. The option `tight` can be used to move the checkboxes to the inner margins of the text block.

Marginal pushbuttons. When `\editSourceOn` is in effect, a pushbutton appears to the right of the marginal checkbox. The action of this pushbutton is to open the source file in the default editor. Modify the appearance using the `\editSourceBtn` command:

```
\editSourceOn                                  (required for the button to appear)
\useEditBtn                                    (the default)
\editSourceBtn[⟨opts⟩]{⟨wd⟩}{⟨ht⟩}
```

The first line `\editSourceOn` is required for the button to appear; usually this command is expanded in the preamble, but it can be expanded in the body of the document to turn on or off (`\editSourceOff`). The second line specifies that pushbutton form field should be used (the default). The third line is the general syntax; here, ⟨*opts*⟩ are key-values that are passed to the underlying `\pushButton` command of eforms.

```
\editSourceBtn[\TU{View in default editor}\S{S}]{11bp}{11bp}   (the default)
```

The above is the default definition for the marginal link.

`\useEditLnk` and `\useEditBtn` discussed

Marginal links. As an alternative to using marginal pushbuttons, pmdb also provides link annotations. When `\editSourceOn` is expanded, marginal buttons appear in the margin, by default. To obtain link annotations also expand the macro `\useEditLnk` (`\useEditBtn` is the default). Use `\editSourceLnk` to customized the link:

```
\editSourceOn                                  (required for the link to appear)
\editSourceLnk
\editSourceLnk[⟨opts⟩]{⟨wd⟩}{⟨ht⟩}{⟨txt⟩}
```

where ⟨*opts*⟩ are key-values that are passed to the underlying link command `\setLink` of eforms. The dimensions provided should be the same as those used by the marginal checkboxes so they are properly aligned.

```
\editSourceLnk[\linktxtcolor{red}\H{N}]{11bp}{11bp}{E}   (the default)
```

The above is the default definition for the marginal button.

Pushbuttons. The package defines two push buttons that should be utilized in your DB document. They can be placed at the end of the document, or in a running footer.

```
\displayChoices[⟨opts⟩]{⟨wd⟩}{⟨ht⟩}
\clrChoices[⟨opts⟩]{⟨wd⟩}{⟨ht⟩}
```

The `\displayChoices` command displays the choices made in the console window of AA/AR, while `\clrChoices` clears all the marginal checkboxes. For example,

Display Choices Clear Choices

```
\displayChoices{}{11bp}   \clrChoices{}{11bp}
```

The ⟨*opts*⟩ argument is to modify the appearance of the buttons. There are several supporting, convenience commands associate with `\displayChoices` and `\clrChoices`:

`\displayChoiceCA{`⟨*string*⟩`}`	(Display Choices)
`\displayChoiceTU{`⟨*string*⟩`}`	(Display all choices in the console window)
`\clrChoicesCA{`⟨*string*⟩`}`	(Clear Choices)
`\clrChoicesTU{`⟨*string*⟩`}`	(Clear all checkboxes created by pmdb)

The 'CA' commands place captions on the buttons; the 'TU' commands defines tool tips for the buttons. The strings shown in parentheses to the right are the default declarations for each of the commands.

6. Comments on portability

The functionality of the document (with the exception of the Ctrl+Click feature) is platform independent; however, to be of any value, the DB files must accompany the DB document. As long as all references to DB files are *relative paths* the DB document can be ported elsewhere along with the supporting DB files. Just ZIP the whole folder containing the DB document and all DB files. They can now be moved to another computer system, unzipped, and total functionality attained. For the Ctrl+Click feature to work, the `aeb-reader.js` file must also be installed. However, recall that the use of the marginal edit button or link annotation does not require installation of a JavaScript file.

7. Final comments

The method of producing the checkboxes in the margins work for many of the situations that arise in producing a LaTeX document; the four 'input modes' `\InputParas`, `\InputQuizItems`, `\InputProbs`, and `\InputItems`, however, may fail in some situations. By studying the DTX file perhaps you can create more input modes that solve your problem.

It has been lovely, but now I must return to my retirement. DS