

AcroTeX.Net

The pmdb Package

D. P. Story

Table of Contents

1	Introduction	3
2	Requirements and options	3
3	The DB stage	4
4	The production stage	5
5	Package commands	6
6	Comments on portability	8
7	Final comments	8

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 $\backslash\text{input}$ 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.

How does this package operate? For a document that inputs content using the \LaTeX command $\backslash\text{input}$, the same content can be input using the command $\backslash\text{pmInput}$, a command defined in this package. When content is input by $\backslash\text{pmInput}$, 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* $\backslash\text{input}$ 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 TEX editor. 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.

2. Requirements and options

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 `pmdb`; 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).

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 $\backslash\text{pmInput}$ 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 left. The `tight` option places the checkboxes “tight” up against the text area. Refer to the **Checkboxes** paragraph for more information.

¹The basic concept for this package was suggested to me by Thorsten G.

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

³These options set Boolean switches, $\backslash\text{ifpmdbmode}$ and $\backslash\text{ifpmdbtight}$, the state of these switches can be changed within the body of the document.

Requirements. The eforms package is required for the creation of checkboxes and push buttons.

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 check box;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 \pmInput. For example, ...
\usepackage{exerquiz} % if needed
\usepackage[dbmode]{pmdb}
...
\begin{document}

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

\pmInput{\{path_1\}}

\pmInput{\{path_2\}}
```

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

```

...
\pmInput{(path_n)}

\displayChoices\quad\clrChoices
\end{document}

```

Descriptions of the various commands `\pmInput`, `\InputQuizItems`, `\InputParas`, `\InputItems`, `\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.

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.

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⟩}
```

*spaces in ⟨path⟩
discussed*

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,

```
\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.

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 three ‘input modes’:

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

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.

\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`

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

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.

Push buttons. 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>}

```

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

```
\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`:

<code>\displayChoiceCA{<string>}</code>	(Display Choices)
<code>\displayChoiceTU{<string>}</code>	(Lists all choices in the console window)
<code>\clrChoicesCA{<string>}</code>	(Clear Choices)
<code>\clrChoicesTU{<string>}</code>	(Clears all checkboxes created by <code>pmdb</code>)

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.

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; however, the three ‘input modes’ `\InputParas`, `\InputQuizItems`, and `\InputItems` may fail in some situations. By studying the DTX file perhaps you can create more input modes that solve your problem.

It was lovely, but now I must return to my retirement. \LaTeX