AcroTEX.Net

The datepicker-pro Package

Picking a date using Flash

D. P. Story

Prepared: July 3, 2018

www.acrotex.net
Version 1.3, 2018/07/02

Table of Contents

1. Introduction

Development for this work occurred in July 2013.[1] At that time a cyber-colleague, Günther Füllerer, sent me a *date picker* he had written using form fields. It was a very impressive work but created a large number of form fields, one for each date in a calendar month. However, instead of working on it myself, improving and generalizing it, I decided to write a date picker package using a Flash window to display the dates, and that is what I did. Below is the date picker without any options; it is set up for English-style dates (MM/DD/YYYY).

Enter a date: ☐

```
\datepicker{date1}{1in}{11bp}
```

Each use of the `\datepicker` command creates four PDF objects: (1) a visible text field that displays the formatted date; (2) a visible push button with an icon appearance; (3) a small hidden text field that holds a "shadow" date, more on that later; and (4) a hidden rich media annotation (RMA) which when activated creates a floating window that displays a calendar.

The `\datepicker` command takes four arguments, the first of which is optional. The first argument is used to modify the date picker, the second is a unique name (`date1` in the above example), these are followed by the width and height of the text input field.

Evolution of date picking. It has been several years since I first issued the datepicker-pro package. Over that period, two events impact this package:

1. Adobe has finally provided native support for date picking through its Acrobat Reader DC application (and in Acrobat itself, of course). Experience the date field below.

   ☐

   ```
   \textField[\AA{%
     \AAKeystroke{AFDate_KeystrokeEx("m/d/yy")}
     \AAFormat{AFDate_FormatEx("m/d/yy")}
   }]{goNatDate}{1in}{11bp}
   ```

   This feature has been long in coming and is finally here. Of course, this date picking feature is not supported in any other PDF viewer, but then, these non-Adobe PDF viewers do not support the approach taken by this package either.

   To my knowledge, there is no way to control the design of the calendar that appears as a popup; for example, a calendar in the German language cannot be specified. Going under the assumption that the calendar will be in the language of the Reader DC being used, I downloaded the German version of Reader DC and determined that the calendar is not localized to the viewer language. That's not good.

2. This package uses Flash technology, which is still supported by Adobe, but it has drawn some of the (user-interface) features offered to Acrobat users. Flash is probably not supported by any other PDF viewer.

---

[1] Then I forgot about it.

To take full advantage of this package, the end-user must use Reader DC or higher.

2. Requirements

This package belongs to the high-class family of AeB Pro, hence, the major requirement of this package is that the PDF be created using Adobe Distiller, see Section 2.2 for details. Of course, if you have Distiller, surely you have Adobe Acrobat, which is also required in the workflow.

2.1. LaTeX package requirements

The package builds on packages developed as part of AeB or AeB Pro:

- aeb_pro: supplies code to create icon appearances for the date picker push buttons.
- rmannot: the package for creating rich media annotations (SWF, FLV, etc); rmannot uses the graphicxsp package.
- eforms: used to create Acrobat form buttons and text fields, both hidden and visible. It also supplies JavaScript support as it imports the insdljs package.

2.2. PDF creator requirements

The big restriction on this package is the requirement to use Adobe Distiller (version 6.0 or later). The package supports the creation of PostScript using dvips and dvipsone. These "drivers" are defined through the required package aeb_pro.

dvipsone
dvips

2.3. Transparency requirements

Transparency is only required to make the RMA transparent; unlike a form field, a RMA does not have a hidden property, so we try to make it transparent.

To get the transparency effect, you must distill using a JOBOPTIONS file that supports transparency. The datepicker-pro package comes with an Distiller job options file named `Standard_transparency.joboptions`. Instead of going through the above rather tedious instructions, drop this file in the place where Distiller expects to find its own JOBOPTIONS files.[2]

If you distill without this `Standard_transparency`, you get an error message in the **Distiller** log that reads like this:

```
%%[Error: The PostScript contains Transparency pdfmark, job aborted.]%%
%%[ /AllowTransparency is false in job option settings.]%%
%%[ Error: undefined; OffendingCommand: pdfmark;
    ErrorInfo: Transparency Group ]%%
```

This suggests that you should use a JOBOPTIONS file that supports transparency! Look for the `Standard_transparency.joboptions` file in the `joboptions` folder of this distribution.

---

[2] Go to Settings > Edit Adobe PDF Settings in the Distiller application window, then click the SaveAs button. A Save Adobe PDF Settings As dialog box opens and you can then see where Distiller likes to save its `.joboptions` file. Copy the provided `.joboptions` to the folder and restart Distiller, the `Standard_transparency` should now be visible in the drop down Default Settings list.

2.4. Viewing requirements

As with any document that contains more than just text content, Adobe Acrobat Reader or Adobe Acrobat is required to actually see the effects of the datepicker-pro package. Most third-party PDF readers do not support interactive form fields, rich-media annotations, and Acrobat JavaScript API. Consumers of your document need to use Adobe Reader or Acrobat.

3. The datepicker-pro commands

We document the commands required in the preamble, the configuration file, and one that can be placed in the preamble or the body of the document.

3.1. Commands for the preamble

A typical preamble would contain the necessary packages required by datepicker-pro:

```
\documentclass{article}
\usepackage[%
    driver=dvips,
    web={extended,tight*,usesf},
    eforms,graphicxsp={showembeds}
]{aeb_pro}
\usepackage{rmannot}
\usepackage{datepicker-pro}
```

```
\documentclass{article}
\usepackage[%
    driver=dvips,
    web={extended,tight*,usesf},
    eforms,graphicxsp={showembeds},
    rmannot
]{aeb_pro}
\usepackage{datepicker-pro}
```

The use of the web package is optional. On the left is the traditional way of using the rmannot package, followed by the datepicker-pro package. More recently (2013/08/05), I've made the rmannot an option of the aeb_pro package. Shown on the right is the rmannot option for aeb_pro.

```
\useFLEXVer{3|4}
\definePath{\dppPath}{path}
\setpickerIcon{path}{list}
```

`\useFLEXVer` The `\useFLEXVer` command declare what version of FLEX is to be used,[3] choices are 3 or 4, corresponding to FLEX 3.6 and FLEX 4.5, respectively. I've found the version 4 is very slow in loading; not recommended. If `\useFLEXVer` is not present in the preamble, version 3 is used by default.

`\dppPath` The `\definePath{\dppPath}` charts out the path to the date picker SWF file; there are two versions `datepicker3.swf` and `datepicker4.swf`. The `\definePath` command is defined in eforms is simply allows you construct paths with normally forbidden characters. The `\dppPath` command is expected by datepicker-pro to be defined; otherwise, the file will not compile. The *path* needs to be an *absolute path* on your file system that points to the location of the SWF files. The declaration of the path config file `\dppPath` may be moved to the configuration file, the topic of Section 3.2

`\setpickerIcon` Finally, the `\setpickerIcon` declares the *path* to the date picker icon; this is a rel-

---

[3] Adobe has since re-branded FLEX to Adobe Flash Builder.

ative (or absolute) path and includes the icon file in the path. The argument *list* is a comma-delimited list of names of each of the date picker fields created in the document. If you want to have several date picker fields, some of which have different date picker icons, then you need to declare additional icons using `\setpickerIcon`. The dimensions of the icon are determined by the command declarations `\setpickerIconWidth` and `\setpickerIconHeight`, these are discussed in Section 3.4. These latter two commands may be declared in the preamble or in the body of the document. This package distribution provides to icon files `dp_icon1.pdf`, `dp_icon2.pdf`, …, `dp_icon7.pdf`, found in the `icons` folder.[4]

The following illustrates the above three commands.

```
%\useFLEXVer{4} % 3 or 4
\definePath{\dppPath}{C:/Users/Public/Documents/My TeX Files/%
    tex/latex/aeb/aebpro/datepicker-pro/swf}
\setpickerIcon{../icons/dp_icon2.pdf}{date1,PickADate,GERDate}
```

In this example, there are three date fields created, each of them is given the appearance of the icon file `dp_icon2.pdf`. Or, we can declare,

```
% \dppPath moved to dp-pro.cfg, see Section 3.2.
\setpickerIcon{../icons/dp_icon1.pdf}{date1,PickADate}
\setpickerIcon{../icons/dp_icon2.pdf}{GERDate}
```

where a different icon is used for the `GerDate` date picker.

3.2. The configuration file

Rather than placing the path to the `swf` folder of this distribution in each file, you can place the declared path in the package configuration file, `dp-pro.cfg`. The file is found in the root directory of the package distribution. An example of its contents is given below.

```
%
% datepicker-pro configuration file
%
\definePath{\dppPath}{C:/Users/Public/Documents/My TeX Files/%
    tex/latex/aeb/aebpro/datepicker-pro/swf}
```

Edit this file to reflect the absolute path to the `swf` folder of this distribution on your file system.

3.3. Commands for the body of the document

Within the body of the document, there is only one command, `\datepicker`:

```
\datepicker[options]{name}{width}{height}
```

The *name* argument should consist only of letters or numbers (however, a number must not be the first character in the name); for example, `date1` or `PickADate` is acceptable.

---

[4] My thanks to Jürgen Gilg for providing additional icon files.

The *`name`* argument is used to build names for the components of the date picker field. The input text field, which is set to read only, is named `txt`⟨*name*⟩, the push button is named `btn`⟨*name*⟩, the hidden text field is named `htxt`⟨*name*⟩, and finally, the RMA has a name of *`name`*. The *`width`* and *`height`* arguments are standard to form fields, as defined by eforms. These two dimensions are applied to the input text field only.

Now for the *`options`* argument of `\datepicker`. The options consist of key-value pairs (`key=value`) and are rather numerous.

- **Format date string**
  - `formatstring`: The value of the `formatstring` key uses various combinations of `M`, `D`, `Y`, and possible `E` as well as spaces and delimiters to format the date string. The following table was extracted from the page:

    `http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/mx/formatters/DateFormatter.html`

    Table of Formatting Patterns

Pattern	Examples	Pattern	Examples
`Y`	`YY = 05`	`D`	`D = 4`
	`YYY=2005`		`DD=04`
	`YYY=02005`		
`M`	`M = 7`	`E`	`E = 1`
	`MM=07`		`EE=01`
	`MMM=Jul`		`EEE=Mon`
	`MMMM=July`		`EEEE=Monday`

    One example is `formatstring={EEEE, DD. MMMM YYYY}`. The default formatting string is `formatstring={MM/DD/YYYY}`.

- **Formatting the calendar.** The dateChooser control that is used in the SWF file contains the name of the month at the top and the names of the days of the week just below it. By default, the usual English month names are used (January, February, ...,December) and the usual English days of the week names are used (Sunday, Monday, ...,Saturday). However, these can be changed through the use of `monthnames` and `daynames` key.

  A note on Unicode. Any glyph outside the range of Basic Latin (U0020-U007F), unicode notation may be used. For example the German word for March is März; it must be specified as `M\u00E4rz`, see the example of German dates presented below.

  - `daynames`: The names of the days that appear on the popup calendar. The value of `daynames` is a comma-delimited list of seven names for the days. The default is `daynames={S,M,T,W,T,F,S}`. These labels appear if the `daynames` key is not used.

- `monthnames`: The names of the months that appear on the popup calendar. The value for `monthnames` is a comma-delimited list of twelve names of the months. The default is `monthnames={January,February,...,December}`. These labels appear if the `monthnames` key is not used. See the initial date picker example on page 3 for an example of these default values.

  Other possible values for these, in the English language, are

  ```
  monthnames={Jan,Feb,Mar,Apr,May,June,July,Aug,Sept,Oct,%
      Nov,Dec}.
  daynames={Su,M,Tu,W,Th,F,Sa}
  ```

  There should be no spurious spaces in the values of any of these keys; in the listing above, I've wrapped the months around to a new line, and inserted a comment character (%) to kill any space that may be generated.

  The order must be the first month of the year (January, or the equivalent in another language) and the first day of the week (Sunday, or its equivalent). Some calendars consider Monday as the first day of the week, if this is so, *still* place Sunday, or its local equivalent, as the first entry in the list. A different starting day is set using the `firstday` key. Setting `firstday=1` puts Monday as the first day of the week on the calendar.

- `firstday`: The value of the key `firstday` is an integer, 0...7. The default is 0.

Next is an example of some German names.

```
daynames={So,Mo,Di,Mi,Do,Fr,Sa},
monthnames={Jan,Febr,Mrz,Apr,Mai,Jun,Jul,Aug,Sept,Okt,%
    Nov,Dez},
firstday=1
```

On the German calendar, *Montag* (*Mo*) is the first day of the week.

- **Formatting the Month and Day in the return value.** When the date string is returned to the input text field it is formatted according to the value of the key `formatstring`. The keys is this section are useful only if your `formatstring` contains any of the formatting combinations `MMM`, `MMMM`, `DDD`, or `DDDD`. These indicate you want the month or day word-names returned, rather than a numerical value.

  The four keys below take as a value of comma-delimited list of names.

  - `monthnamesLong`: The long name of the months and is used with `MMMM` formatting pattern.
  - `monthnamesShort`: The short name of the months, used when `MMM` is the formatting pattern.
  - `daynamesLong`: The long names for the days of the week, used for formatting `DDDD`.

  - `daynamesShort`: The short names for the days of the week and is used with `DDD`.

The defaults are the English counterparts: January/Jan, Sunday/Sun.

For the German language, we can set these values as follows:

```
daynamesLong={Sonntag,Montag,Dienstag,Mittwoch,Donnerstag,%
    Freitag,Samstag},
daynamesShort={So,Mo,Di,Mi,Do,Fr,Sa},
monthnamesLong={Januar,Februar,M\u00E4rz,April,Mai,Juni,%
  Juli,August,September,Oktober,November,Dezember},
monthnamesShort={Jan,Feb,Mrz,Apr,Mai,Jun,Jul,Aug,Sep,%
  Okt,Nov,Dez},
```

Again, always list January (*Januar*/*Jan*) and Sunday (*Sonntag*/*So*) first.

- **Positioning the floating window.** These keys determine the positioning of the floating windows when it opens. The default is the center of the window (for versions > 9), for version 9, this key is ignored and the window appears in the upper-right of the application window. See the documentation of the `rmannot` package for more detail.
  - `halign`: The horizontal positioning of the floating window, permissible values are `near`, `center`, and `far`. The default is `center`.
  - `valign`: The vertical positioning of the floating window, permissible values are `near`, `center`, and `far`. The default is `center`.
  - `hoffset`: The amount of horizontal offset from the initial horizontal alignment (`halign`). See the documentation of the `rmannot` package for more detail. Measured in default user space units (pixels). The default is 0.
  - `voffset`: The amount of vertical offset from the initial vertical alignment (`valign`). See the documentation of the `rmannot` package for more detail. Measured in default user space units (pixels). The default is 0.
- **Setting the dimensions of the window.**
  - `widthOfWindow`: The width of the floating window as measured in default user space units (pixels). The default for this package is 180.
  - `heightOfWindow`: The height of the floating window as measured in default user space units (pixels). The default for this package is 180.

Options can be bundled into a command and passed as the first argument, as this example of the German calendar illustrates.

Wählen Sie ein Datum: (EEEE, DD. MMMM YYYY)

The verbatim listing of this last field is given below.

```
\newcommand{\germanDates}
{%
%    formatstring=DD.MM.YYYY,
    formatstring={EEEE, DD. MMMM YYYY},
    daynames={So,Mo,Di,Mi,Do,Fr,Sa},
    monthnames={Jan,Febr,Mrz,Apr,Mai,Jun,Jul,Aug,Sept,%
        Okt,Nov,Dez},
    daynamesLong={Sonntag,Montag,Dienstag,Mittwoch,Donnerstag,%
        Freitag,Samstag},
    daynamesShort={So,Mo,Di,Mi,Do,Fr,Sa},
    monthnamesLong={Januar,Februar,M\u00E4rz,April,Mai,Juni,Juli,%
        August,September,Oktober,November,Dezember},
    monthnamesShort={Jan,Feb,Mrz,Apr,Mai,Jun,Jul,Aug,%
        Sep,Okt,Nov,Dez},
    firstday=1,
    widthOfWindow=200,heightOfWindow=200
}
\begin{flushleft}
Wählen Sie ein Datum:\ \kern1bp
\datepicker[\germanDates]{GERDate}{2in}{11bp}\quad(EEEE, DD. MMMM YYYY)
\end{flushleft}
```

3.4. Commands for the preamble or the body

There are a few other commands to mention, the declaration or definition take effect with the next date picker command `\datepicker`.

```
\dppToolTip{text}
\pickerOpts{key-values}
\pickerInputOpts{key-values}
\renewcommand{\pickersep}{length}   (3pt)
\pickerIconWidth{length}            (10pt)
\pickerIconHeight{length}           (12pt)
```

`\dppToolTip` The `\dppToolTip` command is used to create a tool tip, text the user sees when he/she rolls over the push button icon. The default is

```
\dppToolTip{Date Picker\n Click to toggle open and close\n
    Shift-click to clear and close}
```

When the user click on the icon, and the calendar is not open, the floating window opens. If the window is open and the user clicks on the icon, the window is closed. If the user shift-clicks and the window, the date field is cleared and the window is closed, if open.

`\pickerOpts` The `\pickerOpts` command lets you pass additional options to the RMAs created by this package. The default is `\pickerOpts{}`. See the documentation of the rmannot package for more details.

`\pickerInputOpts` The `\pickerInputOpts` allows you to pass additional options to the input text field. The default is `\pickerInputOpts{}`. See the documentation of the eforms package for details. We illustrate this last feature:

Pick a date: ▭ ▦

`\datepicker{PickADate}{1in}{11bp}`

```
\pickerInputOpts{\BC{red}\textColor{blue}\Q{1}}%
Pick a date: \kern1bp\datepicker{PickADate}{1in}{11bp}
```

Here, we use a red boundary, blue text color, and center the date.

Any changes to `\pickerOpts` and `\pickerInputOpts` are global unless expanded within a group. Of course, you can bring these options back to their defaults by expanding `\pickerOpts{}` and `\pickerInputOpts{}`.

`\pickersep`

The command `\pickersep` sets the amount spacing between the right edge of the input text field and the left edge of the picker push button. The default is `3bp`.

`\pickerIconWidth`
`\pickerIconHeight`

Finally, the command declarations, `\pickerIconWidth` and `\pickerIconHeight`, determine the width and the height of the date picker icon. The defaults for are shown in parentheses above.

That's all for now, I simply must get back to my retirement. DPS