AcroTEX.Net

Enhanced Preview

D. P. Story

dpstory@acrotex.net
Last Revision Date: September 28, 2019
Version 2.0

1. Introduction

Enhanced preview was introduced into eforms, beginning with 2019/06/14. Enhanced preview attempts to typeset into the document the value of the `\CA` key (for push buttons) and the `\V` key for text fields, choice fields, and radio button and check box fields. This is useful for those using a non-conforming PDF reader such as sumatraPDF. The enhanced preview is activated by expanding `\pmpvOn` (**p**oor **m**an's **p**re**v**iew). A summary of the effects is described below.

push buttons The `\CA` entry is always displayed; however, when the *background color* (`\BG`) *is transparent*, the key-value entry generated by `\CA` *is removed*. This is to avoid two overlaying captions, one typeset into the document, the other part of the button appearance. Examples appear in Section 2.1.

text and choice fields The `\V` entry *is set to empty* (when `\pmpvOn` is active), but the value of the `\V` key is typeset into the document; this is to avoid two overlaying values within the field. There is one special case, when the field is *hidden*; in this case, the value of the field is restored. Hidden text fields are used by the acrotex packages to hold information that can later be retrieved. Examples appear in Section 2.2.

check box and radio button fields These two cases are handled similar to **choice fields**. For these types of fields, the values is typically a mark: a check, an cross, a star, and so on. For preview purposes, eforms defines the declarative command `\pmpvMrk` that takes one argument, the mark to be used. The package declares `\pmpvMrk{X}`, another good choice is `\pmpvMrk{$\checkmark$}`. Examples appear in Section 2.3.

`\pmpvCA`, `\pmpvV`

The macros (local to the option list of a form field) `\pmpvCA` and `\pmpvV` hold the preview values of a push buttons, and all other fields, respectively. With respect to the enhanced preview, the local command `\tops`, used within the argument of `\V` or `\CA`, is `\let` to `\texorpdfstring`. Use `\tops` to offer an alternate text to the value of `\CA` of `\V`. The `\V` key of radio button and check box fields do not handle the `\tops` command, the preview mark is determine by `\pmpvMrk`, as describe above.

`\tops`

For example,

```
\pushButton[\CA{\tops{Tap Me}{Push Me}}]{pbFld}{}{11bp}
```

This button will preview with the caption as 'Tap Me', but will appear within a conforming PDF reader as 'Push Me'; however, if the background color is transparent (`\BG{}`), 'Tap Me' will be the (typeset) caption even in a conforming PDF reader. (This is to avoid overlaying captions.) It is important to say that the final document should be compiled with `\previewOff` and `\pmpvOff` opened in Adobe Reader and saved to obtain proper appearances of the form fields.

Important!

The arguments of `\CA` and `\V` should be text, the text can contain TEX formatting commands, such as `\textbf` or `\textit`, if preceded by `\protect`. For example,

`\protect`

```
\pushButton[\CA{\tops{\protect\textbf{Tap Me}}{Push Me}}]{pbFld}{}{11bp}
```

Now, the enhanced preview will read '**Tap Me**'.

2. Examples

In this section, we present some informative examples of enhanced preview. To see the preview, you need compile using pdflatex, lualatex, xelatex, or latex->dvips->(ps2pdf | distiller), then view the resulting PDF in a non-conforming PDF viewer, such as sumatraPDF. If you use a latex/dvips workflow, you can also view the enhanced preview in your DVI-previewer. Try compiling with various combinations of `\previewOn` or `\previewOff` and `\pmpvOn` or `\pmpvOff`.

2.1. Push button examples

Example 1. Push button Push Me

Example 2. Here is a plain push button, similar to what hyperref generates: Push Me. The previous button has a white background color, now we change it to a transparent color Push Me. Can you see the difference in their behaviors, within Adobe Reader (document still compiled with `\previewOn\pmpvOn`)? Notice the 'Push Me' is tightly fit within the bounding box, this can be adjusted by specifying the third argument: Push Me.

Example 3. Push button that uses `\tops` to have an alternate preview caption: Tap Me, this field has a non-transparent background color so it previews with 'Tap Me', but within Adobe Reader the caption is 'Push Me'.

2.2. Text fields and Choice fields

Many PDF previewers, including sumatraPDF,[1] do support the `\V` key, that is, they display the value of the field.

Example 1. A text field, nothing special: , that field has no `\V` key, this one does true value, 'true value' is typeset into the document. In Adobe Reader, with `\pmpvOn`, the typeset preview is seen because the true field value has been set to empty, to avoid overlaying two initial values of the field. Of course, you don't want this, so always compile with `\previewOff\pmpvOff`, and save in Adobe Reader.

Example 2. A text field using `\tops`: preview value (transparent background) and preview value (white background), the white background prevents the typeset 'preview value' from being seen (in Adobe Reader), but since the `\V` is set to empty when the field is visible, in Adobe Reader you don't see the initial value of 'true value'. To see it, you'll have to compile with `\previewOff\pmpvOff` and open the document in Adobe Reader.

Example 3. Formatting the enhanced preview. In the text field, we format the preview value: ***preview value***, again, to see the 'true value' of the

[1] sumatraPDF does not support the `\CA` key of push buttons.

field, compile with `\previewOff\pmpvOff` and view and save in Adobe Reader. Refer to the source file for details of how this formatting occurred.

Example 4. Positioning the preview value. By default, all preview values are centered within the preview rectangle, but, hey, we can shift it.

preview value

preview value

Example 5. There may be an occasion to rotate a text field (or a button), let's do it! To see the 'true value' of the field, compile with `\previewOff\pmpvOff` and view and save in Adobe Reader. Refer to the source file for details of how this formatting occurred.

Example 6. A visible text field loses its value (when `\pmpvOn` is active), but if its hidden it does not. To test this feature we create a hidden field at the end of this sentence. Now we retrieve this value: Get Secrets.

Example 7. Choice fields: combo box and list box:

Combo box: car

List box 1

car

List box: An example of a list box appears in the right margin (List box 1); it is the 'standard' enhanced preview of a list box.

Here's a crazy idea for improving the preview of the list box example, the same principles can be applied elsewhere. On the right (List box 2), we've fenagled the preview to display all the items, the initial value or default value, is seen preceded by an '*'.

List box 2

*car
truck
van
suv
jeep

This last example is optional, you can remove it (as well as remove the package `\usepackage{forms16be}` from the preamble).

Example 7. One of the motivations for enhanced previews is improved preview when unicode is used for captions or values. The following examples are taken from the `forms16be-ef.tex` sample file of the forms16be package. In the fields below, we use unicode strings to set the button caption or the initial value of the text field. This does not look very good in non-conforming PDF viewers. The enhanced preview feature helps quite a considerably. Try viewing the PDF in sumatraPDF with `\pmpvOn` then with `\pmpvOff`.

Text field: $\alpha\cos(\theta)$

Button: $\alpha\cos(\theta)$

Example 8. For multi-line text fields, a long string for the initial value will most likely exceed the width of the field; you can either turn off enhanced preview by passing the key-value pair `\cmd{\pmpvVOff}`, or you can do something crazy:

I was born in the *bitter-cold* winter of '17, my mother...

Example 7. There is another mechanism for formatting the enhance preview value, it is through the `\pmpvFmt` command. In this example, the preview value does not fit into the text field: Initialize this field with a long string

We can fix this by using the techniques illustrated above, which, at times, can be awkward, or we can use `\pmpvFmt`. The argument of `\pmpvFmt` is the enhanced preview value.

Here, we make the font smaller: Initialize this field with a long string by passing `\cmd{\let\pmpvFmt\footnotesize}` in the optional argument.

Another approach is to turn off the enhanced preview completely for this field by passing `\cmd{\pmpvVOff}` through the optional argument. You can also simple say, `\pmpvVOff\textField...\pmpvVOn`. This latter solution is not as workable because when you want to produce the final build, you must search through the document and change all explicit `\pmpvVOn` commands to `\pmpvVOff`, or comment them out.

Reset all fields in this document.

2.3. Check boxes and radio button fields

Example 1. Check box: X (default mark used)

Example 2. Radio button fields: ✓ (`$\checkmark$` used as mark)