AcroTEX.Net

Creating Button Appearances

D. P. Story

The LATEX source file for this AeB Blog is attached to this document. Click to view attachments panel.

http://www.acrotex.net
Published: September 25, 2009

1. Introduction

In this blog article, we present two methods for creating button appearances; one method uses AeB Pro, the second one uses GraphicxSp.

2. Importing Appearances using JavaScript

This method uses the techniques published in an earlier blog. See the AeB Blog and the article titled *Importing and Placing Images using AeB Pro*.

In the preamble, use the `\declareMultiImages` command to list the icon file you wish to import the `path` key, and the `placement`. The value of the `placement` key may be a comma-delimited list of field names you want this icon to populate. In the preamble of this document, we have,

```
\declareMultiImages
{%
    {path=graphics/man1.pdf,placement={Avatar1,[2]Avatar2}}
    {path=graphics/scot.gif,placement={[1]Avatar1,[1]Avatar2}}
    {path=graphics/girl.png,placement={[2]Avatar1,[0]Avatar2}}
}
\begin{docassembly}
\insertPreDocAssembly
\end{docassembly}
```

There is an optional argument that precedes the field name that determines the face of the button the icon is to appear on; the values are `[0]` (default, normal icon); `[1]` (down icon);

and [2] (rollover icon). The optional argument precedes the field name, and is shown in the example above. There must be no space between the optional argument and the field name; if you type "[2] Avatar1", for example, the field name is interpreted as " Avatar1", which is incorrect. The example is shown below.

The verbatim listing of the above push buttons with custom appearance follows:

```
\def\myPresets{\BC{}\BG{}\S{S}\autoCenter{n}
    \FB{true}\I{null}\TP{1}}
\pushButton[\presets{\myPresets}
    \A{\JS{app.alert("Avatar 1")}}]{Avatar1}{.5in}{.5in}\qquad
\pushButton[\presets{\myPresets}
    \A{\JS{app.alert("Avatar 2")}}]{Avatar2}{.5in}{.5in}
```

Note that I've defined a `\presets` macro, `\myPresets`, to conveniently enter standard optional parameters. The meaning of `\FB`, `\I`, and `\TP` are explained in a later section (Section 4). I've determined that when importing icons in this way, there needs to be an `\I` key, which can be set to `null` at the time of creating of the push button, and is later populated by the icon specified in the `\declareMultiImages` command.

How does it work? This method uses JavaScript methods to import the icons, and to place them in the fields specified by `\declareMultiImages`. The macro itself builds up the necessary JavaScript code, and defines the `\insertPreDocAssembly` command that is placed in the `dosassembly` environment. If you are interested, after you latex the document open the file `docassembly.fdf` to view the code generated.

One of the advantages of this method is that the graphic files do not have to be an EPS file, as they do for the method discussed next. The graphic file itself can be most any format, Acrobat, when it imports the icon, converts it to a PDF icon.

3. Creating Appearances using GraphicxSP

This method uses the GraphicxSP package to import, modify, and embed EPS files that can be used as either pictures inserted into the document, or as appearances to a push button.

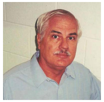
In the preamble, we embed an EPS file

```
\embedEPS[hiresbb,transparencyGroup]{AdobeDon}{graphics/AdobeDon}
```

We can embed as many files as desired.

We can optionally modify an embedded file, here, we set opacity.

```
\begin{createImage}{\bboxOf{AdobeDon}}{nAdobeDon}
    [ {AdobeDon} /SP pdfmark
\end{createImage}
\begin{createImage}{\bboxOf{AdobeDon}}{dAdobeDon}
    [ /ca .3 /SetTransparency pdfmark
    [ {AdobeDon} /SP pdfmark
\end{createImage}
\begin{createImage}{\bboxOf{AdobeDon}}{rAdobeDon}
    [ /ca .5 /SetTransparency pdfmark
    [ {AdobeDon} /SP pdfmark
\end{createImage}
```

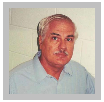
Push button: 

The verbatim listing is

```
Push button: \raisebox{-3pt}{\resizebox{.5in}{!}{\pushButton[%
    \autoCenter{n}\BC{}\S{S}\A{\JS{app.alert("AcroTeX rocks the world!");}}
    \I{nAdobeDon}\RI{dAdobeDon}\IX{rAdobeDon}\TP{1}\FB{true}
]{pbAdobeDon1}{\widthOf{AdobeDon}bp}{\heightOf{AdobeDon}bp}}}
```

Notice we specify values for `\I` (normal appearance), `\RI` (rollover appearance), and `\IX` (down appearance). We use the symbolic names defined earlier by the `createImage` environments. See Section 4 for an explanation for these, as well as for `\TP` and `\FB`.

The eforms package has a user interface mode that uses latex-style key-value pairs. The following is the same example using the `\ui` command.

Push button: 

The verbatim listing is

```
Push button: \raisebox{-3pt}{\resizebox{.5in}{!}{\pushButton[%
    \autoCenter{n}\BC{}\S{S}\A{\JS{app.alert("AcroTeX rocks the world!");}}
    \ui{normappr={nAdobeDon},rollappr={dAdobeDon},
        downappr={rAdobeDon},layout=icononly}
]{pbAdobeDon2}{\widthOf{AdobeDon}bp}{\heightOf{AdobeDon}bp}}}
```

See Section 4 for a description of the parameters used.

The icons created in this way, can be named using the following code.

```
    var f=getField("pbAdobeDon1");
    var nI=f.buttonGetIcon(0);
    var nD=f.buttonGetIcon(1);
    var nR=f.buttonGetIcon(2);
    this.addIcon("nAdobeDon",nI);
    this.addIcon("rAdobeDon",nR);
    this.addIcon("dAdobeDon",nD);
```

This can be executed in the `docassembly` environment.

4. Parameters controlling the Icon Appearance and Placement

The **MK** entry is used to provide an *appearance characteristic dictionary* containing additional information for constructing the annotation's appearance. The eforms package has key-value pairs that populates the **MK** dictionary; we describe the entries in the dictionary, these entries are entered through the optional argument of a `\pushButton` command. In the listing below, we give the key-value pairs, the first is the original key scheme, the second is the more user-friendly key. Additional details can be found in the eForms manual.

I Indirect reference to the normal appearance of an icon. The keys used by eforms are `\I` and `normalappr`.

RI Indirect reference to the rollover appearance of an icon. The keys are `\RI` and `rollappr`.

IX Indirect reference to the down appearance of an icon. The keys are `\IX` and `downappr`.

IF The *icon fit dictionary*. The entries of the **IF** follow:

- **SW** (name; optional) The circumstances under which the icon should be scaled inside the annot rectangle. The key is either `\SW` or `scalewhen`.
  - **A** always scale (the default value).
    KVP: `\SW{A}` or `scalewhen=always`.
  - **B** Scale only when the icon is bigger than the annotation rectangle.
    KVP: `\SW{B}` of `scalewhen=iconbig`.
  - **S** Scale only when the icon is smaller than the rectangle.
    KVP: `\SW{S}` or `scalewhen=iconsmall`.
  - **N** Never Scale.
    KVP: `\SW{N}` or `scalewhen=never`.
- **S** (name; optional) The type of scaling to use the annot rectangle. The key to use is `\ST` or `scale`.
  - **A** *Anamorphic scaling:* Scale the icon to fill the annotation exactly, without regard to the original aspect ratio.
    KVP: `\ST{A}` or `scale=nonproportional`.
  - **P** *Proportional scaling:* Scale the icon to fit the width or height of the rectangle while maintaining the icon's original aspect ratio (ratio width to height) (the default).
    KVP: `\ST{P}` or `scale=proportional`.
- **A** (array; optional) An array of two numbers between 0.0 and 1.0 indicating the fraction of the left over space to allocate at the left and bottom of the icon. A value of

$[\,0.0\ 0.0\,]$ positions the icon at the bottom-left corner; a value of $[\,0.5\ 0.5\,]$ centers it within the rectangle. This entry is only used of the icon is scaled proportionally. The default is $[\,0.5\ 0.5\,]$ the `annot` rectangle. The key is either `\PA` or `position`. The default is `\PA{.5 .5}` (no comma between numbers), in the user friendly style `position={.5 .5}` (no comma between numbers).

FB (Boolean; optional) If `true`, indicates that the button appearance should be scaled to fit fully within the bounds of the annotation without taking into consideration the line width of the border. The default is `false`. The key is `\FB` or `fitbounds`; the default is `\FB{false}` or `fitbounds=false`.

TP a code indicating position of text relative to icon. The key is either `\TP` or `layout`.

0 No icon; caption only. KVP: `\TP{0}` or `layout=labelonly`.
1 No caption; icon only. KVP: `\TP{1}` or `layout=icononly`.
2 Caption below icon. KVP: `\TP{2}` or `layout=icontop`.
3 Caption above icon. KVP: `\TP{3}` or `layout=iconbottom`.
4 Caption to the right of icon. KVP: `\TP{4}` or `layout=iconleft`.
5 Caption to the left of icon. KVP: `\TP{5}` or `layout=iconright`.
6 Caption overlaid on the icon. KVP: `\TP{6}` or `layout=labelover`.

That's it, now, back to my retirement!