



AcroTeX.Net

Creating Custom Stamps with annot_pro

D. P. Story


 The source file for this [AeB Blog](#), taken from the material in [Section 3](#), is attached to this document as a .zip file with a .txt extension, to avoid Acrobat security restrictions. Right-click on the paperclip, and select Save Embedded File to Disc. In the Save File dialog box, change the file name from Sources4cstamp.zip.txt to Sources4cstamp.zip. To use this file, you need a AeB and AeB Pro distribution, as well as the graphicxsp and annot_pro packages.

Table of Contents

1	Introduction	3
2	Defining Custom Stamps through the User Interface	3
3	Defining Custom Stamps with Graphicxsp	6

1. Introduction

The `annot_pro` package for \LaTeX is used to create selected annotations; these are text (or sticky notes), stamp, and file attachment annotations. The home page for `annot_pro` is at http://www.math.uakron.edu/~dpstory/annot_pro.html where you will find the demo file `annots.pdf` that highlights the features of `annot_pro`. To use `annot_pro`, you must have a recent version of Acrobat (Pro), say, version 6.0 or later, and use Adobe Distiller to create PDF files.

This article covers some of the more technical aspects of custom stamp creation. There are two ways to create a custom stamp:

1. By using the Acrobat user interface.
2. By using the `graphicxsp` package.

Each of these two methods are discussed in the sections below.

2. Defining Custom Stamps through the User Interface

Through the user interface of Acrobat Pro, you can create your own custom stamp.

➤ Creating a Custom Stamp

1. On the menu toolbar, go to Tools > Comment & Markup > Stamps > Create Custom Stamp. See [Figure 1](#), page 4.¹
2. In the Select Image for Custom Stamp dialog box, browse and select an image to be used for a stamp. Click OK when you have selected an image.
3. In the Create Custom Stamp dialog box, see [Figure 2](#) on page 4, select a Category from the drop-down menu, or type in the name of a new Category. In the Name field, enter a name for your new stamp. When finished, click OK.
4. Verify that your stamp is listed. Go to Tools > Comment & Markup > Stamps, and select the stamp category you placed your stamp in. You should see your new stamp.

You can avoid using the menu system by opening the Comment & Markup toolbar by selecting Tools > Comment & Markup > Show Comment & Markup Toolbar.

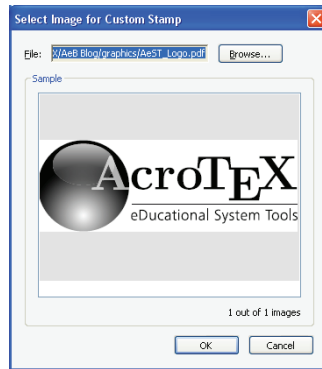


Figure 1: Select Image for Custom Stamp

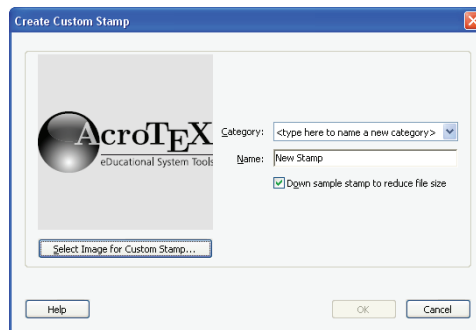


Figure 2: Create Custom Stamp

Once your custom stamp has been registered by Acrobat, you can use the user interface to create a stamp annotation using your stamp. But that is not the point of this section of this article; we want to reference this stamp from within our \TeX source code. To do that, we need to get the value of the name key (of the `\annotpro` command) for this stamp; Acrobat assigned the name when it registered the stamp in its stamp library.

➤ Finding the Name of your Custom Stamp

1. Open the JavaScript Debugger Console Window (Ctrl+J).
2. Copy the code

```
app.getPath({cCategory:"user", cFolder:"stamps"})
```

¹ Shown in [Figure 1](#) is a stamp I created for demonstration purposes.

into the console window; or click this [link](#) to have JavaScript write the code line to the console.

3. Put your cursor on the code line in the console, and press the `Ctrl+Enter`, or the `Enter` key on the numeric keypad. This action will execute the line. The path to the user stamp folder (the folder where you newly created stamp resides) is the output of this code line. Navigate to this folder.
4. In this folder, you will see one or more PDF files. Open the one that is most recently dated, this is the one that no doubt contains your newly created stamp.
5. **Copy this code** to the console. Now, with the user stamp PDF file as the active document (its on top of any other document that's already opened), execute this code by highlighting the code and pressing either press the `Ctrl+Enter`, or the `Enter` key on the numeric keypad.

The code searches through the document, finds all stamps, and reports back to the console. In my case, I get the following output:

```
User name: "AeST", name=#0Ci6LdDj69tMqDgXQAWTTA,
      width=383.37bp,height=134.24bp
User name: "AeBLogo", name=#bYd3xSHP1SP3LAgRhz6f4C,
      width=20.02bp,height=20.02bp
User name: "DPStory", name=#eW0baLhzPsq5rXKnHnp5cA,
      width=185.76bp,height=185.46bp
```

where I have wrapped the output around to a new line so it will fit on the page width. Look at the first entry. In addition to the user name, which we don't use, the JavaScript brings back the value of the `name` parameter to be used with the `\annotpro` command; also returned is the width and height of the stamp. Very cool! These two are also used.

Now that we have all the information we need, we can use `\annot_pro` to create a custom stamp, we'll use the one whose user name is "AeST". Here is the code,

```
1 \annotpro[type=stamp,name=\#0Ci6LdDj69tMqDgXQAWTTA,
2   width=383.37bp,height=134.24bp,widthTo=2in]{Annot_pro rocks!}
```

Code Comments: In line (1) we start the `\annotpro` command with optional parameters of `type=stamp`, followed by the `name` parameter, the one given above. Note that instead of typing `#0Ci6LdDj69tMqDgXQAWTTA`, I've typed `\#0Ci6LdDj69tMqDgXQAWTTA`. The `#` character needs to be escaped when it appears in the optional argument of `\annotpro`. In line (2), we set the natural bounding box of the stamp, these are values returned by the JavaScript, and set `widthTo=2in`, to obtain a stamp 2 inches wide.

The Results of this \LaTeX Code:  **AcroTeX**
eDucational System Tools

I did say we don't use the user interface name, but that wasn't quite true. The `\annotpro` command has a `presets` key that can be used to simplify the creation of this famous stamp. We can define,

```
\newcommand{\myAeST}{type=stamp,
  name=\#0Ci6LdDj69tMqDgXQAWTTA,width=383.37bp,height=134.24bp}
```

then we can simply say.

```
\annotpro[presets=\myAeST,widthTo=2in]{Annot\_pro rocks!}
```

The Results of this \LaTeX Code:  **AcroTeX**
eDucational System Tools

The same results, but with less pain.

➤ **How to create a dynamic stamp.** Lori DeFurio, of Adobe Systems, has a very fine [tutorial and movie posted](#) on this topic, on [AcrobatUsers.com](#).

Important: When using the stamps of Acrobat, or your own stamps through the Acrobat stamp library, always perform a **SaveAs** on the file when you have finished building the file. This imports the appearances of the stamps into the document and saves them.

3. Defining Custom Stamps with Graphicxsp



D. P. Story

You can use any EPS file to create a custom stamp without the user interface, that's what us \LaTeX users like, right? The requirements for this technique is the use of the `graphicxsp` package; if opacity is desired, it is necessary to distill with the **Standard_transparency** job options. (This latter job options file is distributed with the `graphicxsp` package.) In the margin is an example of a custom stamp with `opacity=.5`.



Validate action

One of the reasons for writing this package is for use with the [AcroTeX PDF Blog](#). Using `annot_pro`, I can include code samples using the personal stamp annotation, as shown in the margin.

How to you create these stamps "on the fly?"

➤ **The Preamble.** The following is a minimal preamble of needed packages and commands:

```

1 \usepackage{graphicxsp}
2 \usepackage{annot_pro}
3 ...
4 \embedEPS[hiresbb,transparencyGroup]{AdobeDon}{graphics/AdobeDon}
5 \embedEPS[hiresbb,transparencyGroup]{APBLogo}{graphics/APB_Logo_Sphere}
6 \embedEPS[hiresbb,transparencyGroup]{AeSTLogo}{graphics/AeST_Logo}
7 ...
8 \makeStamp{0.0 0.0 \widthOf{AdobeDon} \heightOf{AdobeDon}}{AdobeDon}{%
9   [ {AdobeDon} /SP pdfmark
10  }
11 \makeStamp{0.0 0.0 \widthOf{APBLogo} \heightOf{APBLogo}}{APBLogo}{%
12   [ {APBLogo} /SP pdfmark
13   }
14 \makeStamp{0.0 0.0 \widthOf{AeSTLogo} \heightOf{AeSTLogo}}{AeSTLogo}{%
15   [ {AeSTLogo} /SP pdfmark
16   }

```

Comments: In lines (1)–(3) we embed EPS files we wish to use in this document. (See the graphicxsp reference manual for a description of this command.) The `\makeStamp` is a special command defined in the `annot_pro` package designed for creating stamps using this technique.

Once we have performed these steps, we can reference this stamp using the `customStamp` and `ap` keys of `annot_pro`.

Now, most of the work has been done. The code for the first of the two custom stamps already visible in the margins are

```

1 \annotpro [type=stamp, customStamp=MyDPSImage, ap=AdobeDon,
2   opacity=.5, widthTo=20bp, margin, readonly, color=blue
3   margintext={\centering\makebox[0pt][c]{D. P. Story}}
4   ]{This is the best picture ....}

```

Comments: The key-value pairs that are new can be found in line (1). We set `customStamp=MyDPSImage`, the value `MyDPSImage` can be any string—you should probably stick to simple letters—the name should not conflict with any name already known to Acrobat. For the value of the `ap` key we use `AdobeDon`, this is a reference to the image by that same name embedded in the document.

The other stamp is done the same way,

```

1 \annotpro [type=stamp, customStamp=MyAeBLogo, ap=APBLogo, widthTo=20bp,
2   readonly, margin, margintext={\centering Validate action},
3   color=blue]{\vbtum}

```

Comments: The command `\vbtum` in line (3) is a macro that holds the verbatim text that appears in the popup window. It was created by the `defineJS` environment of the `insdljs` package.

Finally, for the file labeled `AeSTLogo` that I have embedded in this document, we can display that stamp.



The verbatim listing is

```
1 \annotpro[type=stamp,customStamp=MyAeSTLogo,ap=AeSTLogo,  
2      widthTo=.67\linewidth,color=webbrown]{Logo of AcroTeX.Net}
```

This is the same stamp as created with the user interface, but we created the stamp not by using the techniques of '[Defining Custom Stamps through the User Interface](#),' page 3, but by using the `graphicxsp` techniques, as explained in this section.

Important: Stamps created using `graphicxsp` do not obey the `rotate` key of the `\annotpro` command.

A demo file for this section is an attachment of this document. Save it, unzip it, and try it. See the cover page for the link to the source files.

Well, that's pretty much it for now, I simply must get back to my retirement. ☺