The PDFAnim package

Jochen Skupin

June 22, 2004

Abstract

A LaTeX package to create animated PDFs with pdfLaTeX. This package is still in beta status but I've used it for various presentations for one year now. Maybe it is already useful for other people.

Contents

1 Introduction

PDFAnim is a LaTeX package to create animated PDFs with pdfLaTeX. This package is still in beta status but I've used it for various presentations for one year now. Maybe it is already useful for other people.

As it is very unusual to create animations with LaTeXand to have to deal with JavaScript it might be that large parts of this document are not understandable for the normal LaTeXuser. Don't be to frustrated about it and just try the examples.

At the moment I'm stuck at some points how to improve the implementation of PDFAnim and could need some help how to solve them. That's the reason why I'm releasing it in this early beta state (which holds for the `pdfanim.sty`-file as well as for this documentation). Comments and bug reports are therefor *very* welcome. You can contact me at:

jochen.skupin@uni-bremen.de

Don't be to disappointed if it takes a while till you get an answer but I've only little spare time to work on this package.

2 Usage

To include an animation in a PDF file you need:

- PDFLaTeX
- the PDFAnim package (i.e. the `pdfanim.sty`-file)
- The frames of the animation as single PDF files with filenames in the form `filename`n`.pdf` where n is the number of the nth frame of the animation starting with 0.
- Adobe Acrobat or AcroReader to view the animation

The package includes the single frames as hidden picture buttons in the PDF-file. The animation itself is created using a visible picture button whose picture is exchanged using the JavaScript language available in Adobe Acrobat and AcroReader. The frames of the animation have to be included in the document preamble using

```
\PDFAnimLoad[options]{name}{filename}{number_of_frames}
```

where `filename` is the head of the filename of the PDF-files containing the single frames of the animation omitting the running number and the `.pdf` extension. `name` is the name of the animation. With this name it can be accessed in the document. `number_of_frames` gives the number of files to be included. An animation with 10 frames would require `number_of_frames` = 10 and files `filename0...9.pdf`. The various `options` of `\PDFAnimLoad` will be discussed in section 4.
To access the animation anywhere in the document use:

```
\PDFAnimation{name}
```

Examples can be found at:

- `http://www.uni-bremen.de/~skupin/pdfanim/Demos`
- `http://www.tug.org/texshowcase/`

3 Package option(s)

The package has to be loaded in the document preamble with:

```
\usepackage[options]{pdfanim}
```

The package `options` change the way the JavaScript code is included in the PDF-file.

I would like to get rid of these option because their only purpose is to circumvent problems with the embedding of JavaScript into the PDF-file. They all have advantages and disadvantages:

`default` The default behaviour (by giving no options at all) is to include most of the JavaScript on document level and access it via page open/close attributes or via the on-click action of the picture button.

- The advantage is that only very short function calls have to be included in the page open/close attributes because most JavaScript code is stored on document level. Also it is possible to define global variables to store the status of an animation (like the last displayed frame, is it running or not).
- The disadvantage is that I didn't manage to convince Adobe Acrobat and AcroReader to execute this document level JavaScript when the PDF-file is opened via the command line or with double click on the filename. It has to be opened via the "file open" dialog. This might be OK for personal use in presentations but is rather disturbing for public distributed files.

`NoDocJS` With this option no document level JavaScript is generated. All JavaScript is stored in the picture button itself and the page open/close attributes.

- The advantage is that the embedded JavaScript is executed by Adobe Acrobat and AcroReader regardless how the file is opened.
- The disadvantage is that all JavaScript code is stored on every page again. This leads to slightly bigger file sizes. Due to this all variables are newly initialised on every page so it is not possible to have global variables to remember the status of an animation from page to page.

`NoPageJS` I didn't managed yet to derive on which page a distinct animation (especially floats) is placed by TEX. The JavaScript code for all animations is therefor included in every page. Not very elegantly – I know. The problem is that when the document contains auto starting animations all of them are started again on *every* page. With the `NoPageJS` option the automatic inclusion of JavaScript code on any page can be suppressed.

- The advantage is that not all auto start animations will run permanently.
- The disadvantage is that the user has to take care to put the JavaScript on pages with animations by hand. To do this the following 4 commands are provided:

`\PDFAnimJSPageEnable`	enable JavaScript in page-attribute of current page
`\PDFAnimJSEnable`	enable JavaScript in page-attribute on all following pages
`\PDFAnimJSPageDisable`	disable JavaScript in page-attribute of current page
`\PDFAnimJSDisable`	disable JavaScript in page-attribute on all following pages

4 PDFAnimLoad options

There are already much to much options. But during my usage of PDFAnim I needed all of them for one or the other reason. Unfortunately not all options work well together and some of the older ones might not work at all in the current implementation anymore. Maybe some clean-up is needed here.

4.1 Boolean options

`auto`	auto start animation
`debug`	enable debug messages
`fallback`	include start-picture below PictureButton as fallback solution for xpdf, macs ... (gives poor animation results)
`hidden`	create hidden PictureButton (mostly for internal use)
`loop`	loop animation
`noclick`	don't recognise clicks on PictureButton
`remember`	remember last displayed picture when changing to another page
`reverse`	play animation in reversed order
`step`	advance animation on every mouse-click

4.2 Options that take parameters

`bcolor`	bordercolor of PictureButton
`bgcolor`	backgroundcolor of PictureButton
`defaultframe`	select frame to display when animation not yet running
`depth`	depth of PictureButton
`extension`	set extension of included pictures (till now only pdf works)
`height`	height of PictureButton
`interval`	set interval in ms between animation frames (only shows effect if interval is longer than the time needed to display a frame)
`name`	name PictureButton (may be useful for further editing of the pdf, not needed/used by PDFAnim)
`onclick`	JavaScript action to perform on mouse-click (mostly for internal use)
`scale`	scaling of picture used in PictureButton: A Always scale. B Scale only when the icon is bigger than the annotation rectangle.

S Scale only when the icon is smaller than the annotation rectangle.

N Never scale.

`scaletype`	type of scaling of picture used in PictureButton: A Anamorphic scaling: scale the icon to the annotation rectangle exactly, without regard to its original aspect ratio (ratio of width to height). P Proportional scaling: scale the icon to t the width or height of the annotation rectangle while maintaining the icon s original aspect ratio. If the required horizontal and vertical scaling factors are different, use the smaller of the two, centering the icon within the annotation rectangle in the other dimension.
`use`	use pictures from other animation (to save memory)
`usecnt`	use counter from other animation
`startframe`	select first frame to display
`width`	width of PictureButton

5 Open issues

Include complete animations embedded in a single PDF-file

It would be nice to be able to include complete animations embedded in a single PDF-file instead of having every frame in a single file. Should be possible but I don't know enough about the structure of PDF-files and how to access single objects.

Less options

During my usage of PDFAnim I used all of the options in section 4. But I guess they can be reduced and combined in a better way. Any improvements are welcome.

Document level JavaScript

I'm still looking for a solution how to embed document level JavaScript without having to load the file via the "Open file" dialog. I know that this can be done using the full Adobe Acrobat (demonstrated in the AcroTeX bundle: `http://www.math.uakron.edu/~dpstory/webeq.html`) but it is not my intention to have to use commercial software.

Maybe this is even a Acrobat or AcroReader bug and not solvable by the PDFAnim package.

Navigation buttons

Some users might want to have navigation buttons. This can be done in principle but in my opinion these things destroy a good layout. So I'm not to eager to implement them but if the demand is high ...