

AcroTeX.Net

**The yt4PDF Package**  
**Playing YouTube Videos in PDF**

**D. P. Story**

## Table of Contents

<b>1 Introduction</b>	<b>3</b>
1.1 Sample files . . . . .	3
<b>2 Requirements</b>	<b>3</b>
<b>3 Configuring your installation</b>	<b>4</b>
<b>4 The Rich Mediation Annotation for YouTube</b>	<b>4</b>
4.1 The yt4pdf Poster . . . . .	5
4.2 Control Buttons . . . . .	6
4.3 Controls for playing a YouTube video . . . . .	7
• Playing a video through a link . . . . .	7
• Playing a video using a combo box . . . . .	8
<b>5 Language localizations</b>	<b>9</b>

## 1. Introduction

I believe that I was vaguely aware of the [YouTube ActionScript 3.0 Player](#), but never pursued it because of my general lack of interest in YouTube ([www.youtube.com](http://www.youtube.com)). Recently, I came across a very nice demo PDF that used the API, see [PDF Tube - YouTube API wrapper for PDF documents](#). The SWF file and JavaScript are made freely available, so I downloaded it, and was sufficiently impressed to port the example to  $\LaTeX$  using various members of the  $\AcroTeX$  Fine Family of Software :-). The SWF file and JavaScript are due to the good folks at [UVSAR](#) ([www.uvsar.com](http://www.uvsar.com)). The [UVSAR's](#) sample demo also appeared in the forums of the [Acrobat User Community](#).

The `yt4pdf` package uses the `rmannot` to embed the SWF file that plays the YouTube videos. There is a complete set of buttons that come with the package to control the video: play, pause, stop loading, rewind, load a video by its YouTube video ID, and a button to view a video on the YouTube web site.<sup>1</sup>

### 1.1. Sample files

The following are the sample files shipped with `yt4pdf`:

- `yt4pdf-1.tex` demonstrates the basic functionality of the YouTube RMA: the use of the poster, including `\ytComboList`, `\ytComboBtn`, and `\ytvId`.
- `yt4pdf-2.tex` experiments with different ways the YouTube annotation can be displayed
  - As a small icon, videos are played in a floating window, buttons displayed in the running footer.  
This one uses a rollover, popup menu system (as opposed to a combo box). The `popupmenu` package is used.
  - Using a built-in poster, with the combo box under the annotation, control buttons in running footer.
  - Same as above, but control buttons moved to a two column format.
  - Using the custom YouTube for PDF poster with combo box and control buttons under the RMA.
- `yt4pdf-3.tex` illustrates the usage of the `play` and `load` keys that are passed in the optional parameter of `\ytRmAnnot`.

Advanced examples [AeB Blog site](#).

## 2. Requirements

This package is part of [AeB Pro](#), which means Acrobat Distiller is used to create the PDF; the package requires `rmannot`, which creates rich media annotations. Therefore, we require

**Adobe Acrobat, version 9.0 or later**

<sup>1</sup>Some videos do not allow embedded playback, and must be viewed on the YouTube web site.

To use this package, the document author must have AeB and AeB Pro installed, as well as rannot. The manual for rannot needs to be read closely to properly install it and to function correctly.

### 3. Configuring your installation

In addition to configuring the rannot package correctly, the yt4pdf package needs configuring as well. The yt4pdf package comes with a configuration file yt4pdf.cfg. Open this file in your favorite editor to see

```
% yt4pdf config file. Delete the \endinput below and replace the path
% provided with the path to the swf folder of your yt4pdf installation.
% This path is used to locate the pdf_tube_basic.swf file, used to play
% YouTube videos.
\endinput
\renewcommand{\ytFolder}{C:/Users/Public/Documents/My TeX Files/%
tex/latex/aeb/aebpro/yt4pdf/swf}
```

Edit this file so that \ytFolder points to the swf folder on your computer.

### 4. The Rich Mediation Annotation for YouTube

The Rich Media Annotation (RMA) used is \ytRmAnnot, its definition uses the command \rmAnnot, which is defined in the rannot package.

```
\ytRmAnnot[<kv-pairs>]{<name>}{width}{height}
```

#### Parameter Description:

1. The first (optional) parameter is used to pass the key-value pairs of the \rmAnnot command. Additionally, there are two other kv-pairs recognized, specialized to the \ytRmAnnot command.
  - play=<vID> Loads and plays the YouTube video with video ID of <vID>.
  - load=<vID> Loads the specified video's thumbnail (as specified by <vID>) and prepares the player to play the video. The player does not request the FLV until the play button is pressed.

When you want to play a YouTube video when the page containing the annotation is opened, use the play key; for example play=GZ9e3Dy7obA causes the video whose video ID is GZ9e3Dy7obA to play when the page is opened.

See the manual for the rannot package for more details on the key-value pairs that can be passed through the optional first parameter.

2. The <name> parameter is a name used to refer to the annotation through underlying JavaScript code. The <name> should consist of only letters and numbers. The value of this parameter is placed in the macro \ytCurrRMAName. This macro is used internally by the various control buttons.
3. The width of the annotation.
4. The height of the annotation.

**Annot Dimensions.** The width and height of the YouTube annotation should be 571bp and 330bp, respectively, or any re-scaling of these dimensions.

Examples,

```
\ytRmAnnot[posternote=AcroTeX PDF Tube,play=eNzrn8-JFSE,
  deactivated=pageclose]{YouTube1}{571bp}{330bp}
```

is a “full-size” annotation, while

```
\ytRmAnnot[posternote=AcroTeX PDF Tube,load=eNzrn8-JFSE,
  deactivated=pageclose]{YouTube1}{571bp/2}{330bp/2}
```

is half that size. The first example above plays a video when the page is opened, the second example loads the thumbnail, and waits play the video.

You can also re-scale using either `\resizebox` or `\scalebox` from the `graphicx` package.

```
\resizebox{2in}{!}{\ytRmAnnot[posternote=AcroTeX PDF Tube,
  deactivated=pageclose]{YouTube1}{571bp}{330bp}}
```

or, using `\scalebox`,

```
\scalebox{.5}{\ytRmAnnot[posternote=AcroTeX PDF Tube,
  deactivated=pageclose]{YouTube1}{571bp}{330bp}}
```

**Running multiple videos at once.** At the time of this writing, the **yt4pdf** package is not designed to run more than one video at a time, that’s not saying it cannot be done, however. It is not recommended, indeed, why would you want to look at two YouTube videos at the same time?

For documents with multiple YouTube annotations over several pages it is recommended, therefore, that the `deactivated=pageclose` should be used in the optional parameter list of the `\ytRmAnnot` command.

#### 4.1. The `yt4pdf` Poster

The `rmannot` package provides a standard poster (a gray rectangle with text in the lower left corner). However, there is an especially designed poster, designed by the folks at **UVSAR**. (See Figure 1, page 6.)

**To use this really cool poster:**

1. Insert the following line into the preamble,
 

```
\makePoster[bb=0 0 570.794 329.887]{pdfyt_poster}{pdfyt_poster}
```

 The optional parameter should remain as it is, it adjusts the bounding box so that the poster is properly placed.
2. Use the `poster` key in the first optional parameter of `\ytRmAnnot`, like so,
 

```
\ytRmAnnot[poster=pdfyt_poster,
  deactivated=pageclose]{YouTube}{571bp}{330bp}}
```

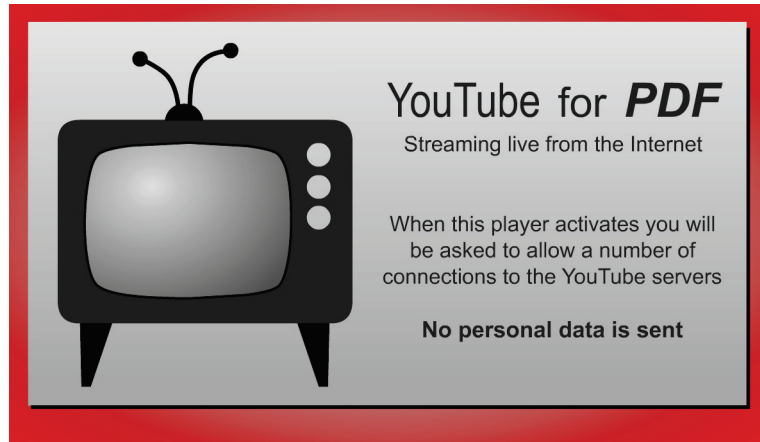


Figure 1: YouTube for PDF poster

#### 4.2. Control Buttons

The YouTube rich media annotation (as created by `\ytRmAnnot`), comes with six (optional) control buttons.

```
\ytPlayToggle[<kv-pairs>]{<width>}{<height>}
\ytStopLoading[<kv-pairs>]{<width>}{<height>}
\ytMuteToggle[<kv-pairs>]{<width>}{<height>}
\ytRewind[<kv-pairs>]{<width>}{<height>}
\ytSelect[<kv-pairs>]{<vID>}{<width>}{<height>}
\ytOpenWeb[<kv-pairs>]{<width>}{<height>}
```

**Parameter Description:** The labels are self-explanatory, the `<kv-pairs>` parameter is the standard eforms optional parameter for changing the appearance of the form. The `<vID>` parameter for `\ytSelect` is a YouTube video ID.

**Command Description:** We give brief descriptions of each of the six control buttons.

- `\ytPlayToggle`: This button has several functions, PAUSE, PLAY, BUFFERING, and REPLAY. Initially the button label is on PAUSE (pressing it pauses a playing video); the button label reads PLAY when the video is loaded and paused (pressing it plays or resumes the video); while the video is loading, the button label is BUFFERING.... After the video finishes playing, the button label is changed to Replay.
- `\ytStopLoading`: Pressing this button stops the streaming of the currently playing video. If the video is started again, there will be some buffering before the play begins.
- `\ytMuteToggle`: This button toggles and sound off and on. Button labels are MUTE and UNMUTE.

- `\ytRewind`: If the video is playing, this button stops the video, rewinds it to the beginning, and starts play again. If the video is paused, this button rewinds the video to the beginning, the video remains paused.
- `\ytSelect`: When this button is pressed, a response dialog opens. The user enters a YouTube video ID into the field. On pressing OK, this video is played. The second parameter `<vID>` is the default video that is displayed in the input box of the response box that opens.  
**Note:** Some YouTube videos are security restricted and cannot be played through an embedded player, such as the one used here. If video ID of a video that does not allow embedded playback, a dialog appears giving the user the opportunity to view the video on the web.
- `\ytOpenWeb`: The user has the option of pressing this button to view the current video on the web at `www.youtube.com`.

The first (optional) parameter can be used to modify the appearance of the individual buttons. To modify the appearances of the buttons as a group, use

```
\ytBtnPresets{<kv-values>}
```

The key-value pairs are ones defined for form fields created by the `efrms` package. The default definition is

```
\ytBtnPresets{\BC{.5 0 0}\S{S}\textFont{HeBo}}
```

The second parameter of `\ytRmAnnot` is the name to be associated with that annot; the name of the annot is not known until after the annot is defined. The command `\DeclareYTName`

```
\DeclareYTName{<name>}
```

allows you to define the name prior to the creation of the annot. This is useful if you want to place the control buttons above the annot itself. By specifying the name, the control buttons will know the name of the annot they are to control.

### 4.3. Controls for playing a YouTube video

To actually play a video, you must pass a video ID to the underlying SWF file. We present two methods here, through links and through a combo box. One of the example files, `yt4pdf-2.pdf`, illustrates another method using a popup menu.

#### • Playing a video through a link

The command `\ytvId` creates a link such that when you click on it, the video specified by the second parameter of the command is played in the associated Rich Media annotation created by `\ytRmAnnot`. The associated RMA is the one most recently defined prior to the link. If the link occurs before the target RMA, you can use `\DeclareYTName` to set the name of the target RMA for the link.

```
\ytvId[<kv-pairs>]{<vID>}{<text>}
```

**Parameter Description:** The first parameter takes the usual key-value pairs for links created by the `eforms` package. The second parameter is the video ID to be played. The third parameter is the text to be displayed by the link, usually the title of the YouTube video.

`\ytvId` loads and plays the specified video; there is an `*`-form of this command,

```
\ytvId* [<kv-pairs>] {<vID>} {<text>}
```

When this form is used, the YouTube thumbnail image for the video is displayed, the player to play the video. The player does not request the FLV until the play button is pressed.

You can set the appearances for all links created by `\ytvId` by using `\ytvIdPresets`

```
\ytvIdPresets {<kv-values>}
```

The key-value pairs are ones defined for form fields created by the `eforms` package. The default definition is

```
\ytvIdPresets {\linktxtcolor{webbrown}}
```

#### • Playing a video using a combo box

You can create a combo box with a play list. There are three commands that create the combo list, `\ytPlayList`, `\ytComboList`, and `\ytComboBtn`.

```
\ytPlayList {<default_vId>} {<playlist>}
```

This command sets the play list to be displayed in the combo box, it needs to be executed before the actual creation of the combo box (using `\ytComboList`). The first parameter is the default video ID, this title will be initially displayed by the combo box. The `<playlist>` is an array of video IDs and title pairs. The following example illustrates the format for the array:

```
\newcommand{\playList}{%
  [(GZ9e3Dy7obA)(Kung-Fu Fighting (Bruce Lee version))]
  [(eNzrn8-JFSE)(Open at Adobe)]
  [(5y9-EVmreU4)(Lori's Corner: Episode \#1)]
}
```

The format for each pair is `[(<vId>)(<title>)]`.

In the example above, the array, in its correct format, is loaded conveniently into a macro `\playList`. The macro may then be passed as the second parameter of `\ytPlayList`. Thus, an example of the use of `\ytPlayList` is

```
\ytPlayList {eNzrn8-JFSE} {\playList}
```

The syntax for the combobox itself is



```
\ytComboList[<kv-pairs>]{<width>}{<height>}
```

The `\ytComboList` is a combo box of video Ids and titles. The user selects a video based on its title, then presses the PLAY button (see `\ytComboBtn` below). This command is executed *before* `\ytComboBtn` to set the play list, and the default value.

**Parameter Description:**

- [`<kv-pairs>`]: (optional) The key-value pairs associated with an eforms form field, used to change the appearance of the field.
- `<width>`: The width of the combo box
- `<height>`: The height of the combo box

The appearance can be changed locally by the optional first parameter. The command `\ytComboListPresets` is used to change all combo lists create by `\ytComboList`. The command takes has one parameter, the key-value pairs for changing the appearance. The default definition is `\ytComboListPresets{}`.

Once the selection has been made from the combo box, the user can play the selection by pressing the button created by `\ytComboBtn`.

```
\ytComboBtn[<kv-pairs>]{<width>}{<height>}
```

**Note:** If you hold down the shift key and click on the button, the thumbnail poster is loaded, and the player is ready to play the video.

**Parameter Description:**

- [`<kv-pairs>`]: (optional) The key-value pairs associated with an eforms form field, used to change the appearance of the field.
- `<width>`: The width of the combo box
- `<height>`: The height of the combo box

The appearance can be changed locally by the optional first parameter. The command `\ytComboBtnPresets` is used to change all combo lists create by `\ytComboBtn`. The command takes has one parameter, the key-value pairs for changing the appearance. The default definition is `\ytComboBtnPresets{}`.

## 5. Language localizations

The default language for yt4pdf is English; the English strings that yt4pdf uses are listed below. These can be redefined as desired.

```
\newcommand{\ytStrPLAY}{PLAY}
\newcommand{\ytStrREPLAY}{REPLAY}
\newcommand{\ytStrPAUSE}{PAUSE}
\newcommand{\ytStrBuffering}{Buffering...}
\newcommand{\ytStrStopLoading}{Stop Loading}
```

```
\newcommand{\ytStrMUTE}{MUTE}
\newcommand{\ytStrUNMUTE}{UNMUTE}
\newcommand{\ytStrREWIND}{REWIND}
\newcommand{\ytStrLoadVideo}{Load Video}
\newcommand{\ytStrWatchOnWeb}{Watch on YouTube}
% response dialog box associated with ytSelect()
\newcommand{\ytRespQues}{Enter the YouTube Video ID}
\newcommand{\ytRespTitle}{Load a Video}
\newcommand{\ytRespLabel}{ID:}
% alert error messages
\newcommand{\ytStrErrorVideoNotFound}{Error: Video not found}
\newcommand{\ytStrErrorNoEmbed}{Error: This video does not allow
    embedded playback \r\r Would you like to view this video
    on the web?}
\newcommand{\ytStrConnectTimedOut}{Connection timed out, try again.}
```

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