rmannot: Support for FLV, SWF, and MP3 in Acrobat 9 Pro

D. P. Story
Email: dpstory@acrotex.net

processed January 30, 2021

Contents

```
1 ⟨*package⟩
```

1 Introduction

The rmannot package was written, in part, to support the AcroFLeX Graphing package; however, this package has wider application. The rmannot package supports the creation of rich media annotations (RichMedia), and the embedding of

SWF and FLV files in a PDF. SWF animations and FLV video can then be played within PDF viewed within version 9 (or later) of Adobe Reader and Acrobat.

Source material for the creation of this package is *Adobe Supplement to the ISO 32000*, June 2008. This document contains the PDF specification of rich media annotations.

Version 9 or later. Beginning with version 9, Adobe Reader and Acrobat contain a Adobe Flash Player, which will play SWF, FLV, MP3 files, and a number of other formats that need to be H.264 encoded.

After December 2020. Adobe drops its support for Flash player (SWF and FLV files) after December 2020; however, this package still works for H.264 encoded videos, including MOV, MP4, M4V, 3GP, 3G2, F4V. The only audio file format supported is MP3.

On the Topic of 3D. Here is something that I've only just come to realize: If you use the UI, and you create a 3D annotation in Acrobat, then give it a SWF as a resource, the 3D annot gets converted into a Rich Media annotation. Looking through the specification as described in the *Adobe Supplement to ISO 32000*, I determined to implement this feature, and why not since most of the structure (rich media annot) was already in place by way of my rmannot package. So, this version of rmannot supports what I'll call Rich Media 3D annotations (RM3DA).

Alexander Grahn, he's the man!

Initially, it was not a challenge to get a 3D model to appear in a RMA created by rmannot, some straight forward modifications to rmannot were required, following ISO 32000. Looking at Alexander Grahn's very fine and brilliant movie15 package, I saw the difficulties of defining and creating *view*s through the LaTeX interface. With Alexander's permission, I gently lifted all the really heavy code from movie15, and placed it in rmannot. I offer up my great and humble thanks for his kindness in allowing the use of his code (characterized by commands beginning with `@MXV`).

```
\RequirePackage{xkeyval}
\RequirePackage{ifpdf}[2006/02/20]
\RequirePackage{ifxetex}[2006/08/21]
\let\rm@One=1 \let\rm@Zero=0
```

(2020/08/21) We test for non-pdfmark drivers, if present, we make minimal package definitions, define all relevant commands to display their ⟨*text*⟩ argument. In this way, pdflatex, lualatex, and xelatex can be used to preview the document, perhaps viewing the results in SumatraPDF.

```
\ifpdf
  \let\RM@action\endinput
\else
  \ifxetex
    \let\RM@action\endinput
  \else
    \let\RM@action\relax
  \fi
\fi
```

```
\ifx\RM@action\endinput
```

We provide definitions to all essential commands and environments to, as best as we can, give a PDF that is viewable, but with no functionality.

```
\RequirePackage{eforms}[2020/12/14]
\DeclareOptionX*{}
\ProcessOptionsX\relax
%\let\AcroVer\@gobble
\newcommand\AcroVer[2][]{}
\let\saveNamedPath\@gobbletwo
\def\rma@edefexecute#1{\edef\rm@@temp@@exp{#1}\rm@@temp@@exp}
\newcommand\makePoster[3][]{}
\define@key{rmAnnot}{width}{\def\rmAnnot@width{#1}}
\define@key{rmAnnot}{scale}{\def\rmAnnot@scale{#1}}
\let\rmAnnot@width\@empty
\define@key{rmAnnot}{height}{\def\rmAnnot@height{#1}}
\let\rmAnnot@height\@empty
\newcommand{\rmAnnot}[4][]{%
  \bgroup
    \setlength{\dimen@}{#2}\xdef\rm@Annot@width{\the\dimen@}%
    \setlength{\dimen@}{#3}\xdef\rm@Annot@height{\the\dimen@}%
  \egroup
  \rma@edefexecute{\noexpand\setkeys*{rmAnnot}{#1}}%
  \bgroup
  \ifx\rmAnnot@width\@empty
    \ifx\rmAnnot@height\@empty
    \else
      \setlength{\dimen@}%
        {\rmAnnot@height*\ratio
          {\rm@Annot@width}{\rm@Annot@height}}%
      \xdef\rm@Annot@width{\the\dimen@}%
      \setlength{\dimen@}{\rmAnnot@height}%
      \xdef\rm@Annot@height{\the\dimen@}%
    \fi
  \else
    \setlength{\dimen@}%
      {\rmAnnot@width*\ratio
        {\rm@Annot@height}{\rm@Annot@width}}%
    \xdef\rm@Annot@height{\the\dimen@}%
    \setlength{\dimen@}{\rmAnnot@width}%
    \xdef\rm@Annot@width{\the\dimen@}%
  \fi
  \egroup
  {\previewOn\pushButton[\CA{Distiller required}\BC{}\BG{}
  \S{S}\Ff{\FfReadOnly}]{btn}{\rm@Annot@width}{\rm@Annot@height}}}
\def\setRmOptions3D#1#2{}
\PackageWarningNoLine{rmannot}
  {PDF creation requires Adobe Distiller.\MessageBreak
  Workflow is latex > dvips > distiller; otherwise,\MessageBreak
  this package does nothing}
```

```
\fi
\RM@action % \endinput or \relax
```

2 Options of this package

use3D When the use3Doption is invoked, the annot3d.def code file is input at the end of the package, and the fp package is loaded, is package is used to calculate the views matrices.

```
\DeclareOptionX{use3D}{%
  \def\rma@input@iiidCode{\InputIfFileExists{annot3d.def}{}{}}%
  \def\rma@requirefp{\RequirePackage[nomessages]{fp}}%
}
\let\rma@input@iiidCode\relax
\let\rma@requirefp\relax
```

Process options, there can be only one!

```
\ProcessOptionsX
```

We use graphicxsp to generate annotation appearances.

```
\RequirePackage{graphicxsp}
\rma@requirefp
\RequirePackage{ifthen}
```

3 Preliminary Code

A counter to track the annots as they are created.

```
\newcounter{rm@Cnt}
```

Some switches and markers to prevent embedding the same file multiple times.

```
\newif\ifrma@EmbedFile\rma@EmbedFiletrue
\newif\ifrma@EmbedVideoPlayer\rma@EmbedVideoPlayerfalse
\let\rma@isVPEmbedded\rm@Zero
\newif\ifrma@EmbedAudioPlayer\rma@EmbedAudioPlayerfalse
\let\rma@isAPEmbedded\rm@Zero
```

We use a utility command, taken and renamed from the comment package.

```
\def\rm@csarg#1#2{\expandafter#1\csname#2\endcsname}
```

\pathToSkins
\pathToPlayers

For FLV files, one of standard skins are used to control the play. We need to know where the skins are located on the system (for distiller) and where the players are. Use \pathToSkins to specify the location of the skins, and \pathToPlayers. Specifying the \pathToSkins also defines the path to players. Currently, the players are in a sub-folder of the Multimedia Skins folder. We include \pathToPlayers in case future releases move the players elsewhere; in this case, \pathToPlayers must be executed after \pathToSkins.

```
\newcommand{\pathToSkins}[1]{\begingroup
  \Hy@unicodefalse\pdfstringdef\rma@pathToSkins{#1}%
  \gdef\PathToSkins{\rma@pathToSkins}%
  \gdef\rma@pathToPlayers{\rma@pathToSkins/Players}\endgroup
}
```

\AcroVer (2015/09/30) Added \AcroVer and a more intelligent method of finding the path to skins and players. The optional argument of \AcroVer takes key words win or mac, the default is win. Typically, the argument of \AcroVer is a number, 9, 10, 11, but beginning with the DC versions is can be 2015 (classic) or DC (subscription).

```
\newif\ifuseWinAcrobat\useWinAcrobattrue
\define@choicekey{rmAcroVer}{win}{32,64}[32]{%
  \appType{#1}\useWinAcrobattrue}
\define@key{rmAcroVer}{mac}[mac]{\useWinAcrobatfalse}
```

The syntax is \AcroVer[win|mac]{ver}

```
\def\appType#1{\def\@rgi{#1}\def\@tstii{64}%
  \def\p@thHash{ (x86)}\ifx\@rgi\@tstii\let\p@thHash\@empty\fi
}
\def\p@thHash{ (x86)}
\newcommand{\AcroVer}[2][]{%
  \def\rmDC{DC}\def\rmBeta{Beta}\def\rmArgi{#1}%
  \def\AcrobatVer{#2}\ifx\rmArgi\@empty\else
    \setkeys{rmAcroVer}{#1}\fi\def\@x{\string\ }%
```

If this is the DC version, we handle appropriately

```
  \ifx\AcrobatVer\rmDC
    \ifuseWinAcrobat
      \edef\rmSkinPath{C:/Program Files\p@thHash/Adobe/Acrobat DC/%
        Acrobat/Multimedia Skins}\else
      \edef\rmSkinPath{/Applications/Adobe{\@x}Acrobat{\@x}DC/%
        Adobe{\@x}Acrobat.app/Contents/Resources/%
        Multimedia{\@x}Skins}\fi
  \else\ifx\AcrobatVer\rmBeta
    \ifuseWinAcrobat
      \edef\rmSkinPath{C:/Program Files\p@thHash/Adobe/Acrobat Beta/%
        Acrobat/Multimedia Skins}\else
      \edef\rmSkinPath{/Applications/Adobe{\@x}Acrobat{\@x}Beta/%
        Adobe{\@x}Acrobat.app/Contents/Resources/%
        Multimedia{\@x}Skins}\fi
```

If \AcrobatVer is not DC or Beta it is a number. Possible values are 9, 10, 11, 2015, 2016,...

```
  \else
    \ifnum\AcrobatVer<9\relax
      \PackageError{rmannot}{Acrobat version 9 or later
      supports\MessageBreak rich multimedia annotations}
      {Upgrade your Acrobat to a more recent version.}%
    \else
      \ifuseWinAcrobat
```

We are on a Windows OS machine

```
        \ifnum\AcrobatVer<12\relax
```

When version is less than 12, the version numbers are decimal numbers, `9.0`, `10.0`, `11.0`. We append '`.0`' to the end of `\AcrobatVer`.

```
     \edef\rmSkinPath{C:/Program Files (x86)/Adobe/%
      Acrobat \AcrobatVer.0/Acrobat/Multimedia Skins}%
    \else
```

When version is greater than 12, the version is a year 2015, etc.

```
     \edef\rmSkinPath{C:/Program Files (x86)/Adobe/%
      Acrobat \AcrobatVer/Acrobat/Multimedia Skins}%
    \fi
   \else
```

We are on a Mac OS machine

```
    \ifnum\AcrobatVer<12\relax
     \ifnum\AcrobatVer=9\relax
```

Special path for version 9

```
      \edef\rmSkinPath{/Applications/%
       Adobe{\@x}Acrobat{\@x}\AcrobatVer{\@x}Pro/%
       Adobe{\@x}Acrobat{\@x}Pro.app/Contents/%
       MacOS/Multimedia{\@x}Skins}%
     \else % ver 10 or 11
```

Versions 10 and 11 are referred to using roman numerical numbers (`X` and `XI`)

```
      \ifnum\AcrobatVer=10\relax\def\romanVer{X}\else
      \ifnum\AcrobatVer=11\relax\def\romanVer{XI}\fi\fi
      \edef\rmSkinPath{/Applications/%
       Adobe{\@x}Acrobat{\@x}\romanVer{\@x}Pro/%
       Adobe{\@x}Acrobat{\@x}Pro.app/Contents/%
       Resources/Multimedia{\@x}Skins}%
     \fi
    \else
```

`\AcroVer` is greater than 12, it must be a year, 2015, 2016, etc.

```
     \edef\rmSkinPath{/Applications/%
      Adobe{\@x}Acrobat{\@x}\AcrobatVer/%
      Adobe{\@x}Acrobat.app/Contents/Resources/%
      Multimedia{\@x}Skins}%
    \fi
 \fi\fi\fi\fi
 \expandafter\pathToSkins\expandafter{\rmSkinPath}%
}
\@onlypreamble\AcroVer
\AcroVer{DC}
\newcommand{\pathToPlayers}[1]{\pdfstringdef\rma@pathToPlayers{#1}}
```

`\defineRMPath` A simple command for defining paths. We use `\hyper@normalise` (from hyperref). Special characters are made safe to use. The command takes two arguments, the first is the control sequence of the path you want to define; the second argument is the path. For example,

```
\defineRMPath{\myURLRMFiles}{http://www.example.com/~dpspeaker/videos}
```

```
\newcommand{\defineRMPath}[1]{\def\rm@ctrlName{#1}%
  \hyper@normalise\rm@defineURLPath}
\def\rm@defineURLPath#1{\expandafter\xdef\rm@ctrlName{#1}}
```

\useVideoPlayerPlus

(2010/09/29 v1.0b) Added support for the use of a video player with additional features that are nice. VideoPlayerPlus.swf is supplied by Joel Geraci, see his blog article at blogs.adobe.com/pdfdevjunkie/2010/03/introducing_the_video_player_p.html.

(2016/10/09) Removed support for VideoPlayerPlus.swf; this widget is no longer supporter by the author of the widget. All functionality of the player plus widget is included in the player X widget.

```
\newif\ifVideoPlayerEx\VideoPlayerExfalse
\def\rma@VideoPlayer{VideoPlayer.swf}
\newcommand{\useVideoPlayerPlus}{%
  \PackageWarning{rmannot}{The \string\useVideoPlayerPlus\space
    is no longer supported,\MessageBreak
    will use \string\useVideoPlayX\space instead.
    In the future\MessageBreak
    specify \string\useVideoPlayX}%
```

Use VideoPlayerX.swf instead.

```
  \useVideoPlayerX
}
```

\useVideoPlayerX

We also support the video player begin developed (and in beta) by UVSAR at http://www.uvsar.com/projects/acrobat/videoplayerx/.

```
\newcommand{\useVideoPlayerX}{\VideoPlayerExtrue
  \def\rma@VideoPlayer{VideoPlayerX.swf}%
  \let\useVideoPlayerPlus\relax
}
```

\useVideoPlayerPlus and \useVideoPlayerX allowed only in the preamble.

```
\@onlypreamble\useVideoPlayerPlus
\@onlypreamble\useVideoPlayerX
```

Javascript API for Multimedia. We present some convenience macros for controlling the multimedia players; we support two players VideoPlayer.swf (Adobe) and VideoPlayerX.swf (UVSAR). We shall refer to these three as **VPB** and **VPX**, respectively.

Core API. Valid for all supported.

```
\def\mmPlay{"multimedia_play"}
\def\mmPause{"multimedia_pause"}
\def\mmRewind{"multimedia_rewind"}
\def\mmNextCuePoint{"multimedia_nextCuePoint"}
\def\mmPrevCuePoint{"multimedia_prevCuePoint"}
\def\mmSeek{"multimedia_seek"}
\def\mmMute{"multimedia_mute"}
\def\mmVolume{"multimedia_volume"}
```

VPX API. The following are defined for **VPX**. The JavaScript API supported by UVSAR's `VideoPlayerX.swf`. Some API are more advanced and not listed, others—listed under the core above—change the return value from `void` to a non-`void` return. Full documentation for these is at `http://www.uvsar.com/projects/acrobat/videoplayerx/`.

```
\def\mmSource{"multimedia_setSource"}
\def\mmSkin{"multimedia_setSkin"}
\def\mmSkinColor{"multimedia_setSkinColor"}
\def\mmSeekCuePoint{"multimedia_seekCuePoint"}
\def\mmSkinAlpha{"multimedia_setSkinAlpha"}
\def\mmGetSource{"multimedia_getSource"}
\def\mmUseLocal{"multimedia_useLocal"}
\def\mmGetMetaData{"multimedia_getMetdata"}
\def\mmGetVideoState{"multimedia_getVideoState"}
\def\mmSetScaleMode{"multimedia_setScaleMode"}
\def\mmGetVersion{"multimedia_getVersion"}
```

Version 10.2 of **VPX**, **VPX** now recognizes API of **VideoPlayerPlus**, we therefore include `\mmSkinAutoHide`; hence, there is no reason to even use **VideoPlayerPlus**, which we now don't anymore.

```
\def\mmSetStageColor{"multimedia_setStageColor"}
\def\mmIsLooping{"multimedia_isLooping"}
\def\mmSkinAutoHide{"multimedia_skinAutoHide"}
```

Version 10.4 of **VPX** added the following function.

```
\def\mmShowLoopButton{"multimedia_showLoopButton"}
```

The **VPX** is a superset of the API of the core and **VPPlus**; there is actually no reason to use the **VPPlus** anymore.

Usage: The following example sets the source for the RMA to play.

```
var rm=this.getAnnotRichMedia(this.pageNum, "myRMA");
rm.callAS(\mmSource, "myVideo");
```

The code is valid for **VPX**, for the basic **VPB**. Extensive examples may be found on the **AeB Blog** `http://blog.acrotex.net`.

Other utility macros

```
\def\ps@mark{[\space}
\def\rma@edefexecute#1{\edef\rm@@temp@@exp{#1}\rm@@temp@@exp}
```

These label commands were taken from movie15, needed for comparability with of the movie15 code being used in the 3D portion of this package.

```
\def\@MXV@newlabel#1#2{{%
  \rm@csarg\xdef{#1}{#2}}}%
\def\@MXV@getlabelvalue#1{%
  \rm@csarg\ifx{#1}\relax%
    undefined%
  \else%
    \csname#1\endcsname%
  \fi%
}%
```

Macro for writing labels to external *.aux file

```
\def\@MXV@labeltoaux#1#2{%
  \@bsphack\protected@write\@auxout{}{%
    \string\@MXV@newlabel{#1}{#2}%
    \string\@MXV@newlabel{@#1@}{\@MXV@getlabelvalue{#1}}%
  }\@esphack%
  \ifthenelse{%
    \equal{\@MXV@getlabelvalue{#1}}{undefined}\or%
    %double check that the value hasn't changed
    \not\equal{\@MXV@getlabelvalue{#1}}{\@MXV@getlabelvalue{@#1@}}%
  }{%
```

Issue warning only once, at end of document

```
    \ifthenelse{\isundefined{\@MXV@warning}}{%
      \gdef\@MXV@warning{}%
      \AtEndDocument{%
        \PackageWarningNoLine{rmannot}{%
        @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\MessageBreak
        @@ Rerun to get object references right! @@\MessageBreak
        @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@}%
      }%
    }{}%
  }{}%
}%
```

\RefObjRm holds the indirect reference to the annotation with name of #1, used with RM3DA in /GoTo3DView

```
\def\RefObjRm#1{{\@MXV@getlabelvalue{rmAnnot_#1}}}
```

Configuration File. We supply a configuration file, after the definition of \pathToSkins to to read in the path to the skins on the local file system.

```
\InputIfFileExists{rmannot.cfg}{}{}
```

\saveNamedPath
\rma@useNamedPath

Paths to SWF and FLV files can be saved under unique names with the command \saveNamedPath. The first parameter is a symbolic name (unique) and is used in the fourth argument of \rmAnnot and as values of the resources key. The path is sanitized using the hyperref command \hyper@normalise.

When we are processing a MP3 file, we need to embed the poster image only once. the following command embeds the poster, then redefines itself to \relax. This macro is used in \rm@saveNamedPath.

```
\def\rma@embed@mpiii@Poster{%
  \embedEPS[hiresbb]{ramp3poster}{ramp3poster}%
  \global\let\rma@embed@mpiii@Poster\relax
}
```

\audCtrlWd
\audCtrlHt

The default legacy dimensions of the MP3 control, these are and .

```
\def\audCtrlWd{613bp}\let\cntrlbrWd\audCtrlWd
\def\audCtrlHt{66bp}\let\cntrlbrHt\audCtrlHt
```

This is the MP3 poster image, but we'll only use it once. The command redefines itself to `\relax`.

```
\def\rma@set@mpiiiposter{%
  \begin{sp@createImage}{\bboxOf{ramp3poster}}{nramp3poster}%
      \rma@invisible
      \ps@mark{ramp3poster} /SP pdfmark
  \end{sp@createImage}%
  \global\let\rma@set@mpiiiposter\relax
}
```

`\saveNamedPath` **Syntax:** `\saveNamedPath[<MimeType>]{<name>}{<path>}`

(12/27/10) The default for `[#1]` was `\rma@mimetype@swf`, have now changed this to `\@empty`.

```
\newcommand{\saveNamedPath}[2][]{%
  \edef\rm@argii{#2}\@ifundefined{rma@@#2}%
  {\gdef\rm@thisPath{rma@@#2}}{\rma@PkEr@ii}%
  \gdef\rm@thisMimeType{#1}%
  \hyper@normalise\rm@saveNamedPath
}
\def\rm@saveNamedPath#1{%
  \rm@csarg\gdef{\rm@thisPath}{#1}%
  \rm@csarg\xdef{rma@mt@\rm@argii}{\rm@thisMimeType}%
```

We check to see if this is an MP3 file. If so, we embed the screen shot of the `AudioPlayer` controls as a default poster. The graphic file `ramp3poster.eps` needs to be on the graphics search path of LaTeX.

```
  \rma@edefexecute{\noexpand\filename@parse{#1}}%
  \@ifundefined{filename@ext}{\rma@PkEr@iii{#1}}{}%
  \rma@edefexecute{\noexpand
    \uppercase{\noexpand\def\noexpand\rma@tempi
    {\filename@ext}}}
```

We define the filename with extension under a convenience command. Authors are encouraged to use this command when referencing an embedded file in the `flashvars` key. Its value should be basename.ext, unless part of the path is enclosed in braces, in which case, a folder may be included, for example `assets/myVideo.flv`. This value is the one that appears in the **Resources** tab of the **Edit Flash** under **Name**.

```
  \rm@csarg\xdef{\rm@argii FileName}{%
      \filename@base.\filename@ext}%
  \rm@csarg\xdef{\rm@argii URL}{%
     \filename@area\filename@base.\filename@ext}%
```

Embed the file `ramp3poster.eps`. GraphicxSP required.

```
  \ifx\rma@tempi\rma@rmAnnot@type@mpiii\rma@embed@mpiii@Poster\fi
}
```

`\rma@useNamedPath` is used internally to access the path through its name.

```
\def\rma@useNamedPath#1{\@nameuse{rma@@#1}}
\def\rma@resource#1{\csname#1FileName\endcsname}
\def\rma@urlresource#1{\csname#1URL\endcsname}
```

Within an eforms widget, `\Name` and `\urlName` are defined. This is to make eforms consistent with `\rmAnnot`, where `\Name` and `\urlName` are let to `\rma@resource` and to `\rma@urlresource`, respectively.

```
\expandafter\def\expandafter\makeJSspecials\expandafter{\makeJSspecials
  \let\Name\rma@resource\let\urlName\rma@urlresource
}
```

`\rmaName`
`\rmaUrlName`

Public versions of `\rma@resource` and `\rma@urlresource`. (10/18/2011) Added public versions of `\rma@resource` and `\rms@urlresource` that have parentheses to delimit argument. This way they can be used inside the `insDLJS` env.

```
\let\rmaName\rma@resource
\def\rmaNameP(#1){\rma@resource{#1}}
\let\rmaUrlName\rma@urlresource
\def\rmaUrlNameP(#1){\rma@urlresource{#1}}
```

Supported Extensions & Mime Types. A couple of text macros, used to compare with the extensions provided by the document author. Currently, we support SWF, FLV, MP3 files, and F4V files (Acrobat 10 or later).

(2011/11/03) Added support for MP4, M4V, MOV, 3GP, 3G2, all require H.264 encoding; if these files do not have H.264 encoding, rmannot/distiller will embed anyway, but the video will not play. (2011/12/10) Added U3D and PRC to the list of file types supported.

```
\def\getargsiii#1#2#3{\def\aeb@argi{#1}\def\aeb@argii{#2}%
  \def\aeb@argiii{#3}}
\@tfor\rma@data:={{uiiid}{U3D}{model/u3d}}{{prc}{PRC}{model/prc}}%
  {{swf}{SWF}{application/x-shockwave-flash}}%
  {{flv}{FLV}{video/x-flv}}{{fiv}{F4V}{video/mp4}}%
  {{mpiv}{MP4}{video/mp4}}{{mivV}{M4V}{video/x-m4v}}%
  {{mov}{MOV}{video/quicktime}}{{iiiGP}{3GP}{video/3gpp}}%
  {{iiiGii}{3G2}{video/3gpp2}}{{mpiii}{MP3}{audio/x-mp3}}\do{%
  \expandafter\getargsiii\rma@data
  \rm@csarg\edef{rma@rmAnnot@type@\aeb@argi}{\aeb@argii}
  \rm@csarg\edef{rma@mimetype@\aeb@argi}{\aeb@argiii}
}
```

`VideoPlayer`
`AudioPlayer`

Symbolic name of the video player and audio players.

We delay the definitions of `VideoPlayer`, `AudioPlayer`, and the skins until the beginning of the document, if the user can specify `\useVideoPlayerX` in the preamble, `\rma@VideoPlayer` will expand to VideoPlayerX.swf. Also, the new command `\AcroVer` may be executed in the preamble to set the path to the skins, so we'll delay all `\saveNamedPath` definitions to the beginning of the document.

```
\def\rm@SkinsAndPlayerPaths{%
  \saveNamedPath{VideoPlayer}{\PathToSkins/Players/\rma@VideoPlayer}%
  \saveNamedPath{AudioPlayer}{\PathToSkins/Players/AudioPlayer.swf}%
```

`skin1`
`all`
`skin2`
`skin3`
`skin4`
`skin5`
`skin6`
`skin7`

We predefine the seven skins, these should also be used as resources of `\rmAnnot` when either `VideoPlayerPlus` or `VideoPlayerX` is used, and when the skins are to be changed dynamically.

```
\saveNamedPath{skin1}{\PathToSkins/SkinOverAllNoFullNoCaption.swf}%
\saveNamedPath{all}{\PathToSkins/SkinOverAllNoFullNoCaption.swf}%
\saveNamedPath{skin2}%
    {\PathToSkins/SkinOverAllNoVolNoCaptionNoFull.swf}%
\saveNamedPath{skin3}{\PathToSkins/SkinOverPlay.swf}%
\saveNamedPath{skin4}{\PathToSkins/SkinOverPlayMute.swf}%
\saveNamedPath{skin5}{\PathToSkins/SkinOverPlaySeekMute.swf}%
\saveNamedPath{skin6}{\PathToSkins/SkinOverPlaySeekStop.swf}%
\saveNamedPath{skin7}{\PathToSkins/SkinOverPlayStopSeekMuteVol.swf}%
}
```

Now, make all these path definitions at the beginning of the document.

```
\AtBeginDocument{\rm@SkinsAndPlayerPaths}
```

none
noChange

Two special convenience definitions. We make definitions so that \Name{none} and \urlName{none} expand to \@empty; \Name{noChange} and \urlName{none} both expand to the string noChange.

```
\@namedef{noneFileName}{}
\@namedef{noneURL}{}
\@namedef{noChangeFileName}{noChange}
\@namedef{noChange}{noChange}
```

The following is a convenience text macro, this string appears repeatedly throughout this file.

```
\def\rma@ANT{rmAssetsNameTree-\therm@Cnt}
```

4 The \rmAnnot command

The \rmAnnot command creates a rich media annotation (AnnotRichMedia). Currently, this package supports SWF, FLV, F4V, MP4, M4V, MOV, 3GP, 3G2, and MP3 files. Normally, SWF files are applications that contains their own navigation controls; the FLV and F4V files are played by a VideoPlayer.swf file, shipped with Acrobat 9 Pro, and a controlling skin, also shipped with Acrobat. The \rmannot tries to support most of the features available through the user interface.

Acrobat 9 Pro supports FLV, SWF, and MP3 files as well, Acrobat 9 Pro Extension supports other video formats, by first converting that format to FLV and embedding them in the document. Adobe Flash (CS5) Professional can save a movie as SWF, and the included utility Adobe Flash Video Encoder can convert movie files to the FLV or F4V format.

Beginning with version 9.2, the direct inclusion of videos with extensions of MP4, M4V, MOV, 3GP, 3G2 requires that the fils use the H.264 codec.

3D Support. (2011/12/10) Added U3D and PRC to the list of file types supported; when one of these files appears in #4 of \rmAnnot a RM3DA is created.

Certain features of 3D, specifically **3D measurement** and **3D commenting**, as well the **Select Model**, **Select Face**, and **Select 3 Points** found in the **Camera Properties** dialog box, may or may not work, depending on the information is the 3D file. For example dice.u3d cannot be measured, unless it is imported by

Acrobat, which apparently parses the file to obtain the required information for measuring.

4.1 Options for \rmAnnot

The xkeyval package is used to develop options for the \rmAnnot command.

4.1.1 Annot Name

name Use the name key to specify the annotation name. If not specified, this package generates the name aebRM\therm@Cnt.

```
\define@key{rmAnnot}{name}[aebRM\therm@Cnt]{\def\rma@Annot@name{#1}}
```

4.1.2 Launch Settings

enabled This option determines how the annot is activated, there are three possible values: onclick (activated when user clicks on the annot, or by user script); pageopen (activated when the page is opened); pagevisible (activated when the page becomes visible).

```
\define@choicekey+{rmAnnot}{enabled}[\val\nr]%
 {onclick,pageopen,pagevisible}[onclick]{%
 \ifcase\nr\relax
   \def\rma@rmAnnot@enabled{/XA}\or
   \def\rma@rmAnnot@enabled{/PO}\or
   \def\rma@rmAnnot@enabled{/PV}\fi
}{\PackageWarning{rmannot}{Bad choice for enabled, permissible values
   are onclick, pageopen and pagevisible. Try again}}
```

deactivated This option determines how the annot is de-activated, there are three possible values: onclick (de-activated by user script or by right-clicking on the annot and choosing Disable Content); pageclose (de-activated when the page is closed); pageinvisible (de-activated when the page becomes invisible).

```
\define@choicekey+{rmAnnot}{deactivated}[\val\nr]%
 {onclick,pageclose,pageinvisible}[onclick]{%
 \ifcase\nr\relax
   \def\rma@rmAnnot@deactivated{/XD}\or
   \def\rma@rmAnnot@deactivated{/PC}\or
   \def\rma@rmAnnot@deactivated{/PI}\fi
}{\PackageWarning{rmannot}{Bad choice for deactivated, permissible
    values are onclick, pageclose and pageinvisible. Try again}}
```

windowed When this boolean option is set to true (or is just included in the option list), the video is viewed in a floating window. The default is to view the video within the annot.

```
\define@boolkey{rmAnnot}{windowed}[true]{}
```

When the rich media annotation appears as a floating window, the initial dimensions and position of that window can be set by the following key-value pairs. Use \setWindowDimPos to set all these values.

```
\define@key{winDimPos}{width}{\def\rma@winDimPos@width{#1}}
  \define@key{winDimPosWidth}{default}[288]%
    {\def\rma@winDimPosWidth@def{#1}}
  \define@key{winDimPosWidth}{max}[576]%
    {\def\rma@winDimPosWidth@max{#1}}
  \define@key{winDimPosWidth}{min}[72]%
    {\def\rma@winDimPosWidth@min{#1}}
\define@key{winDimPos}{height}{\def\rma@winDimPos@height{#1}}
  \define@key{winDimPosHeight}{default}[216]%
    {\def\rma@winDimPosHeight@def{#1}}
  \define@key{winDimPosHeight}{max}[432]%
    {\def\rma@winDimPosHeight@max{#1}}
  \define@key{winDimPosHeight}{min}[72]%
    {\def\rma@winDimPosHeight@min{#1}}
\define@key{winDimPos}{position}{\def\rma@winDimPos@position{#1}}
  \define@choicekey+{winDimPosPos}{halign}[\val\nr]%
    {near,center,far}[far]{%
    \ifcase\nr\relax
      \def\rma@winDimPosPos@halign{/Near}\or
      \def\rma@winDimPosPos@halign{/Center}\or
      \def\rma@winDimPosPos@halign{/Far}\fi
  }{}
  \define@choicekey+{winDimPosPos}{valign}[\val\nr]%
    {near,center,far}[near]{%
    \ifcase\nr\relax
      \def\rma@winDimPosPos@valign{/Near}\or
      \def\rma@winDimPosPos@valign{/Center}\or
      \def\rma@winDimPosPos@valign{/Far}\fi
  }{}
  \define@key{winDimPosPos}{hoffset}[18]%
    {\def\rma@winDimPosPos@hoffset{#1}}
  \define@key{winDimPosPos}{voffset}[18]%
    {\def\rma@winDimPosPos@voffset{#1}}
```

`\setWindowDimPos` When the window is floating (`windowed=true`) there are a number of parameters that govern the dimensions and position on page. These key-value pairs are not set up as optional arguments of `\rmAnnot`. Set them in vertical mode, for the next rich media annotation.

```
\providecommand{\setWindowDimPos}[1]{%
  \setkeys{winDimPos}{#1}%
  \edef\temp@expand@sets{%
    \noexpand\setkeys{winDimPosWidth}{\rma@winDimPos@width}%
    \noexpand\setkeys{winDimPosHeight}{\rma@winDimPos@height}%
    \noexpand\setkeys{winDimPosPos}{\rma@winDimPos@position}%
  }\temp@expand@sets
}
```

`\resetWindowDimPos` The command `\resetWindowDimPos` resets the window parameters to their default values.

```
\providecommand{\resetWindowDimPos}{%
  \setWindowDimPos{width={default,max,min},height={default,max,min},
    position={halign,valign,hoffset,voffset}}%
}
\resetWindowDimPos
```

url Use the url key, a boolean, to indicate that the path to the file is a URL.

```
\define@boolkey{rmAnnot}{url}[true]{}
```

4.1.3 \setRmOptions3D: UI for 3D

\setRmOptions3D 3D options for the annot with name of #1; the options are passed by #2. This supports the 3D tab of the user interface for the Edit 3D dialog box.

Proposed syntax:

```
\setRmOptions3D{myDice}
{
  3DOptions={⟨options-from-movie15⟩},
  3DResources={
    none={rName=⟨name1⟩},...,
    foreground={rName=⟨name2⟩,flashvars=⟨vars⟩},...,
    background={rName=⟨name3⟩,flashvars=⟨vars⟩},...,
    material={rName=⟨name4⟩,
              mName=<materialName>,flashvars=⟨vars⟩},...
  }
}
```

When the \rmAnnot does not take a 3D file as its 4th argument, the above options are ignored.

```
\def\setRmOptions3D#1#2{\rm@csarg\xdef{#1_3DOPTS}{#2}}
```

4.1.4 Annot appearance options

borderwidth The width of the border of the annot. Possible values are none (the default), thin, medium, and thick.

```
\define@choicekey+{rmAnnot}{borderwidth}[\val\nr]%
 {none,thin,medium,thick}[none]{%
 \ifcase\nr\relax
   \def\rma@rmAnnot@borderwidth{0}\or
   \def\rma@rmAnnot@borderwidth{1}\or
   \def\rma@rmAnnot@borderwidth{3}\or
   \def\rma@rmAnnot@borderwidth{5}\fi
}{\PackageWarning{rmannot}{Bad choice for borderwidth, permissible
   values are none,thin,medium,and thick. Try again}}
```

poster The name of a embedded graphic to be used as a poster for the video.

posternote If the poster key is not specified, a substitute poster will be generated, see the definition of \defaultPoster. This default poster has a little message, or note, in the lower left corner. The default message is an advertisement for AcroTEX followed by the words Flash, Video, or MP3, depending on the file type.

```
\define@key{rmAnnot}{poster}[]{\def\rma@rmAnnot@poster{#1}}
\define@key{rmAnnot}{posternote}[AcroTeX \rma@poster@descrip]%
  {\def\rma@posternote{#1}}
\define@boolkey{rmAnnot}{defaultposter}[true]{}
```

`invisible` (2010/09/29 v1.0a) When the invisible option is used and there is no poster option, the poster is transparent. This makes it useful when viewing the video in a window, and you want to hide the annot in an obscure corner of the page (or under a form field). In this case, the video/audio is played by JavaScript.

```
\define@key{rmAnnot}{invisible}[]%
  {\def\rma@invisible{\ps@mark/ca 0/SetTransparency pdfmark }}
\let\rma@invisible\@empty
```

`transparentBG` This option is available only for SWF files. Set the background to transparent.

```
\define@boolkey{rmAnnot}{transparentBG}[true]{%
  \ifKV@rmAnnot@transparentBG
    \def\rma@rmAnnot@transparent{true}\else
    \def\rma@rmAnnot@transparent{false}\fi
}
```

`width` `height` We attempt to resize the annot according to the `width` or `height`. Resize proportionally. Only the first one of these two keys is obeyed, never both.

```
\define@key{rmAnnot}{width}{\def\rmAnnot@width{#1}}
\let\rmAnnot@width\@empty
\define@key{rmAnnot}{height}{\def\rmAnnot@height{#1}}
\let\rmAnnot@height\@empty
```

`scale` We attempt to resize the RMA according to the `scale` factor provided. The `scale` key is only obeyed if no `width` or `height` key is specified.

```
\define@key{rmAnnot}{scale}{\def\rmAnnot@scale{#1}}
\let\rmAnnot@scale\@empty
```

`toolbar` `modeltree` These are keys concern 3D annots. `toolbar` is a Boolean, which if true (the default), causes the 3D toolbar to appear when the annot is activated. If `toolbar=false`, the toolbar does not appear when the annotation is activated.

```
\define@boolkey{rmAnnot}{toolbar}[true]{%
  \ifKV@rmAnnot@toolbar
    \def\rma@rmAnnot@toolbar{true}\else
    \def\rma@rmAnnot@toolbar{false}\fi
}
```

`modeltree` is a Boolean, which if true causes the **Model Tree** as viewed in the **Navigation Pane**. The default is false, the **Model Tree** is not displayed when the annotation is activated.

```
\define@boolkey{rmAnnot}{modeltree}[true]{%
  \ifKV@rmAnnot@modeltree
    \def\rma@rmAnnot@modeltree{true}\else
    \def\rma@rmAnnot@modeltree{false}\fi
}
```

passcontext This option is available only for SWF files. SWF file developers can select this option to replace the Acrobat context menu with the context menu of the originating SWF file. When the user right-clicks the SWF file, the available options are from the originating file.

Pass right-click context to Flash. Should be used only if there is a way of deactivating the annotation, perhaps through JavaScript.

```
\define@boolkey{rmAnnot}{passcontext}[true]{%
  \ifKV@rmAnnot@passcontext
  \def\rma@rmAnnot@PassContextClick{true}\else
  \def\rma@rmAnnot@PassContextClick{false}\fi
}
```

4.1.5 Options for the skins

Skins are used with video files.

skin When playing a supported video file, various skins can be used. I've labeled them skin1–skin7, all, and none. The names of the SWF files describe each skin.

Note: By experimenting with the UI, it is apparent that one cannot have different controls for the same video file.

```
\define@choicekey+{rmAnnot}{skin}[\val\nr]%
  {none,all,skin1,skin2,skin3,skin4,skin5,skin6,skin7}[skin1]{%
  \edef\rma@skinName{#1}% 2011/10/18 changed from number to name
  \ifcase\nr\relax
    \let\rma@rmAnnot@Skin\@empty\or % none
    \def\rma@skinName{skin1}%
    \def\rma@rmAnnot@Skin{SkinOverAllNoFullNoCaption.swf}\or % all
    \def\rma@rmAnnot@Skin{SkinOverAllNoFullNoCaption.swf}\or % 1
    \def\rma@rmAnnot@Skin{SkinOverAllNoVolNoCaptionNoFull.swf}\or % 2
    \def\rma@rmAnnot@Skin{SkinOverPlay.swf}\or % 3
    \def\rma@rmAnnot@Skin{SkinOverPlayMute.swf}\or % 4
    \def\rma@rmAnnot@Skin{SkinOverPlaySeekMute.swf}\or % 5
    \def\rma@rmAnnot@Skin{SkinOverPlaySeekStop.swf}\or % 6
    \def\rma@rmAnnot@Skin{SkinOverPlayStopSeekMuteVol.swf}\fi % 7
}{%
  \@ifundefined{rma@@#1}{%
  \PackageWarning{rmannot}{Bad choice for 'skin,' permissible
  values are none, all, skin1--skin7, or a custom skin already
  defined. Try again}}{%
```

If a value of skin is not one of the defaults, we allow the user to define a skin, name and path to a skin SWF must be declared using \saveNamedPath.

```
    \PackageWarning{rmannot}{Recording new skin, '#1'}%
    \edef\rma@skinName{#1}%
    \edef\rma@rmAnnot@Skin{\csname#1FileName\endcsname}%
    \rm@csarg\let{embedSkin#1}\rm@One
  }%
}
```

We use `\let\csname embedSkin<skinName>\endcsname` determining to embed or not to embed a particular skin. We initialize these markers to 1 (embed). After each annot, the marker corresponding to the skin used is set to 0; hence we will not embed it again.

```
\@tfor\rma@arg:={skin0}{skin1}{skin2}{skin3}{skin4}%
  {skin5}{skin6}{skin7}\do{%
  \rm@csarg\let{embedSkin\rma@arg}\rm@One
}
```

`skinAutoHide` A Boolean key that determines if the skin automatically hides itself when the mouse pointer is removed from the annot. The default is `true`.

```
\define@boolkey{rmAnnot}{skinAutoHide}[true]{%
  \ifKV@rmAnnot@skinAutoHide
    \def\rma@skinAutoHide{true}\else
    \def\rma@skinAutoHide{false}\fi
}
```

`skinBGColor` The color of the skin, represented as a hex number, the default is `0x5F5F5F`.

```
\define@key{rmAnnot}{skinBGColor}[0x5F5F5F]%
  {\def\rma@skinBGColor{#1}}
```

`skinBGAlpha` The alpha for the skin, a number between 0 and 1. The default is 0.75.

```
\define@key{rmAnnot}{skinBGAlpha}[0.75]%
  {\def\rma@skinBGAlpha{#1}}
```

`volume` The initial volume of the audio track. Values range from 0 (muted) to 1. The default is 1.0. The volume may be adjusted by the user at run time, if the selected skin has a volume control, or by a JavaScript control.

```
\define@key{rmAnnot}{volume}[1.00]%
  {\def\rma@rmAnnot@volume{#1}}
```

4.1.6 Animation Settings

`speed` Description quoted from the *Adobe Suppl. Doc*: A positive number specifying the speed to be used when running the animation. A value greater than one shortens the time it takes to play the animation, or effectively speeds up the animation.

```
\define@key{rmAnnot}{speed}[1]{%
  \def\rma@rmAnnot@speed{#1}%
}
```

`playcount` Description quoted from the *Adobe Suppl. Doc*: An integer specifying the play count for this animation style. A nonnegative integer represents the number of times the animation is played. A negative integer indicates that the animation is infinitely repeated. The default is -1.

```
\define@key{rmAnnot}{playcount}[-1]{%
  \def\rma@rmAnnot@playcount{#1}%
}
```

4.1.7 Adding Resources

resources

Some SWF files require other files to run. The resources key allows you to list all files that are required to run the SWF file. Currently, the additional resources are other SWF files only. Items must be specified using `\useNamedPath` command, and is a lists of delimited by braces

```
resources={mySWF1,mySWF2}
```

where `mySWF1` and `mySWF2` are defined by `\saveNamedPath`.

```
\newtoks\rma@toks\rma@toks={}
\newcount\rma@nResources
\newif\ifrma@isiiid\rma@isiiidfalse
\define@key{rmAnnot}{resources}[]{\rma@toks={}\rma@nResources=0\relax
  \ifrma@isiiid\let\rma@next\relax
    \else\def\rma@next{\rma@proc@resources{#1}}\fi\rma@next
}
```

When we are doing 3D, we don't process resources through `rmAnnot` family. When we are processing 3D resources, the files are entered through a separate command `\setRmOptions3D`

```
\newcommand{\rma@proc@resources}[1]{%
  \def\rma@rmAnnot@resources{#1}%
  \ifx\rma@rmAnnot@resources\@empty\let\rma@addResources\@empty
  \let\rma@addFileSpecs\@empty\else
```

We process resources when there are some to process `:-)`

```
       \@for\rma@arg:=\rma@rmAnnot@resources\do{%
         \advance\rma@nResources1\relax
         \rma@edefexecute{\noexpand
           \filename@parse{\rma@useNamedPath{\rma@arg}}}%
         \@ifundefined{filename@ext}{%
           \rma@PkEr@iii{\rma@useNamedPath{\rma@arg}}}{}%
         \edef\rma@fs@expand{rmFileStrm\rma@arg}%
         \@ifundefined{\rma@fs@expand}{%
```

If this resource has not been used, we define it, and give it a indirect reference: `rmfstream\therm@Cnt-\the\rma@nResources`

```
           \rm@csarg\xdef{\rma@fs@expand}%
             {rmfstream\therm@Cnt-\the\rma@nResources}%
             \def\rma@embed{1}}{\def\rma@embed{0}}%
```

Add this file to our token list of all resources.

```
         \edef\rma@tmp@exp{\the\rma@toks%
           \noexpand\\{\the\rma@nResources}%
           {\filename@area}{\filename@base.\filename@ext}%
           {\rma@embed}{\csname\rma@fs@expand\endcsname}%
```

(12/27/10) changed the expansion to `\expandafter\noexpand` to leave a token as as this argument. Later, in `\rm@appendFileSpecs` we test this token against `\@empty`. This argument is `#6` in `\rm@appendFileSpecs`.

```
          {\rm@csarg\noexpand{rma@mt@\rma@arg}}}%
          \rma@toks=\expandafter{\rma@tmp@exp}%
        }% do
        \let\\\rm@appendNameTree
        \expandafter\xdef\expandafter\rma@addResources%
            \expandafter{\the\rma@toks}%
        \let\\\rm@appendFileSpecs
        \expandafter\xdef\expandafter\rma@addFileSpecs%
          \expandafter{\the\rma@toks}%
    \fi
}
```

The two commands \rm@appendNameTree and \rm@appendFileSpecs take the same six parameters, here they are

#1 The resource number, typically \the\rma@nResources

#2 \filename@area

#3 \filename@base.\filename@ext

#4 The value of \rma@embed, 1 to embed, 0 for not embedding file has already been embedded.

#5 The indirect reference to the file stream

#6 The mime type of this resource

\rm@appendNameTree

This is used with the token list \rma@toks, before this list is expanded, \\ is \let to \rm@appendNameTree. We create a name tree of additional resources specified by the resources key. This is saved in a macro \rma@addFileSpecs to be used later.

```
\def\rm@appendNameTree#1#2#3#4#5#6{%
  \ps@mark{\rma@ANT} (#3) /APPEND pdfmark^^J%
  \ps@mark{\rma@ANT} {rmfilespec\therm@Cnt-#1}/APPEND pdfmark^^J%
}
```

\rm@appendFileSpecs

This is used with the token list \rma@toks, before this list is expanded, \\ is \let to \rm@appendFileSpecs. We create a set of file specs and file streams, if needed, for each of the resources. This is saved in a macro \rma@addResources to be used later.

```
\def\rm@appendFileSpecs#1#2#3#4#5#6{%
```

File specs. If things work out well, /F {#5} is an indirect reference to the file stream, whether it is a new one that has never been embedded, or is one that has already been embedded.

```
  \ps@mark/_objdef {rmfilespec\therm@Cnt-#1}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmfilespec\noexpand\therm@Cnt-#1} <<%
    /F(#3)%
    /UF (#3)%
    /EF <</F {#5}>>%
    /Type/Filespec%
  >>/PUT pdfmark^^J%
```

File stream: We only embed if `#4` is equal to 1, if it is 0, this particular file has already been embedded.

```
  \if#41
    \ps@mark/_objdef {#5} /type/stream/OBJ pdfmark^^J%
    \ps@mark{#5} (#2#3) (r) file /PUT pdfmark^^J%
    \ps@mark{#5}<<%
      /Type/EmbeddedFile%
```

(12/27/10) The `/Subtype` key is optional, so we make it optional if `#6` is empty.

```
\ifx#6\@empty\else
      /Subtype(#6)%
\fi
      >>/PUT pdfmark^^J%
      \ps@mark{#5} /CLOSE pdfmark^^J%
  \fi
}
```

`flashvars` Flash variables to pass to a SWF file.

```
\define@key{rmAnnot}{flashvars}[]{%
    \def\rma@rmAnnot@flashvars{#1}%
}
```

`cuepoints` Populate the `CuePoint` dictionary of the `Params` dictionary. This one is tricky. There are two types of cue points, Navigation and Event. Navigation corresponds to cue points that have been encoded onto the Flash media; Event corresponds a time value on the media, and is not, I believe, encoded on the media. We need the type (Navigation or Event), the Name, the Time and the action (A).

Event cue points are used to trigger ActionScript methods when the cue point is reached, and let you synchronize the video playback to other events within the Flash presentation.

Navigation cue points are used for navigation and seeking, and to trigger ActionScript methods when the cue point is reached. Embedding a navigation cue point inserts a keyframe at that point in the video clip to enable viewers to seek to that place in the video.

```
\define@key{rmAnnot}{cuepoints}[]{%
  \edef\rma@rmAnnot@cuepoints{#1}%
  \ifx\rma@rmAnnot@cuepoints\@empty\else
    \rma@nResources=0\relax% dps
    \def\rma@array@hold{}\def\rma@dict@hold{}%
    \@for\arg:=\rma@rmAnnot@cuepoints\do{%
      \advance\rma@nResources1\relax
      \rma@edefexecute{\noexpand\setkeys{rmCuePt}{\arg}}%
      % need to build the array of indirect references,
      \edef\rma@array@hold{\rma@array@hold\space
        {rmCuePoints\therm@Cnt-\the\rma@nResources}}%
      % and the code for the cue point dictionary
      \edef\rma@dict@hold{\rma@dict@hold
        \ps@mark/_objdef %
          {rmCuePoints\therm@Cnt-\the\rma@nResources}%
          /type/dict/OBJ pdfmark^^J%
        \ps@mark{rmCuePoints\therm@Cnt-\the\rma@nResources} <<
          /Type/CuePoint
\ifx\rma@rmCuePt@name\@empty
          /Name (RMACP \the\rma@nResources)
\else
          /Name (\rma@rmCuePt@name)
\fi
          /Subtype \rma@rmCuePt@type
          /Time \rma@rmCuePt@time
\ifx\rma@rmCuePt@action\@empty\else
          /A << /Type/Action\JS{\rma@rmCuePt@action} >>
\fi
              >> /PUT pdfmark^^J%
      }%
    }% end of \@for
  \fi
}
```

type (cuepoints)
name (cuepoints)
time (cuepoints)
action (cuepoints)

These are the key-value pairs used for defining cue points, they are `type`, `name`, `time`, and `action`. These keys are entered as a value of the `cuepoints` key, like so,

```
\newcommand{\myCuePoints}{%
 {type=nav,name=Chapter1,time=0,action={console.println("Chapter1")}},%
 {type=nav,name=Chapter2,time=1883,action={console.println("Chapter2")}},%
 {type=nav,name=Chapter3,time=5197,action={console.println("Chapter3")}},%
 {type=nav,name=Chapter4,time=6817,action={console.println("Chapter4")}},%
 {type=nav,name=Chapter5,time=9114,action={console.println("Chapter6")}},%
 {type=nav,name=Chapter6,time=12712,action={console.println("Chapter6")}}
}
```

Note the use of the comment symbol `%` following at the end of each line. We define the cue points using a command, then pass it to `\rmAnnot`, like so,

```
   \rmAnnot[cuepoints={\myCuePoints}]{320bp}{240bp}{sample}
```

Note the parentheses around `\myCuePoints`, the command expands to contain commas, so the rules of xkeyval say to enclose in parentheses.

The structure generated in the PDF file seems to be correct, but we get JavaScript errors. Could this be a bug in Acrobat? For now, this a temporary feature.

```
\define@choicekey+{rmCuePt}{type}[\val\nr]%
    {event,nav}[nav]{%
    \ifcase\nr\relax
        \def\rma@rmCuePt@type{/Event}\or
        \def\rma@rmCuePt@type{/Navigation}\fi
}{\PackageWarning{rmannot}{Bad choice for type, permissible values
   are event and nav. Try again}}
\define@key{rmCuePt}{name}[]{%
```

```
    \def\rma@rmCuePt@name{#1}%
}
\define@key{rmCuePt}{time}[0]{%
    \def\rma@rmCuePt@time{#1}%
}
\define@key{rmCuePt}{action}[]{%
    \def\rma@rmCuePt@action{#1}%
}
```

Default values of key-values pairs. We set the default values of all the keys on startup.

```
\setkeys{rmAnnot}{name,url=false,enabled,deactivated,borderwidth,%
  windowed=false,poster,posternote,skin,skinAutoHide,skinBGColor,%
  skinBGAlpha,volume,speed,playcount,resources,flashvars,%
  transparentBG=false,passcontext=false,cuepoints,toolbar,%
  modeltree=false}
```

4.2 The definition of \rmAnnot

The following two utility commands are used in the definition of \rmAnnot to record the embedding of the video/audio player, and to set switches to prevent the re-embedding of the player again.

```
\def\rma@recordVideoPlayer{%
  \ifrma@EmbedVideoPlayer
    \global\let\rma@isVPEmbedded\rm@One\global\rma@EmbedVideoPlayerfalse
  \else
    \if\rma@isVPEmbedded\rm@Zero
      \global\rma@EmbedVideoPlayertrue\fi
  \fi
}
\def\rma@recordAudioPlayer{%
  \ifrma@EmbedAudioPlayer
    \global\let\rma@isAPEmbedded\rm@One\global\rma@EmbedAudioPlayerfalse
  \else
    \if\rma@isAPEmbedded\rm@Zero
      \global\rma@EmbedAudioPlayertrue\fi
```

If the file is MP3, and there is no poster defined for it, we use the default MP3 poster, a screen shot of the AudioPlayer controls.

```
    \ifx\rma@rmAnnot@poster\@empty\rma@set@mpiiiposter\fi
%     \rma@set@mpiiiposter
  \fi
}
```

A switch to indicate success or failure when looking for a Video extension. Used in \rmAnnot.

```
\newif\if@FndSuppExt \@FndSuppExtfalse
```

\rmAnnot The \rmAnnot command creates a rich media annotation. This package supports file types SWF, FLV, F4V, MP4, M4V, MOV, 3GP, 3G2, and MP3, the latter being a sound format. The command takes four parameters, the first optional, the others required.

[#1]: Insert key-value pairs for changing the appearance, launch, and control settings

#2: The width of the annot

#3: The height of the annot

#4: The name of the file to play (the name is defined by the `\saveNamedPath` command)

```
\newcommand{\rmAnnot}[4][]{\begingroup
  \PMPV{#4}%
```

We `\let` `\Name` to `\rma@resource`. The author can then refer to the name of the resource within, for example, the `flashvars` key.

```
  \let\Name\rma@resource
  \let\urlName\rma@urlresource
  \makeJSspecials
```

Empty these to macros for they might contain content from a previous annot with resources.

```
  \let\rma@addResources\@empty\let\rma@addFileSpecs\@empty
```

We begin by passing the dimensions through a length so the author can use the `calc` package.

```
  \bgroup
    \setlength{\dimen@}{#2}\xdef\rm@Annot@width{\the\dimen@}%
    \setlength{\dimen@}{#3}\xdef\rm@Annot@height{\the\dimen@}%
  \egroup
```

Next we increment a running counter, to give each annot, and all indirect references a unique name.

```
  \stepcounter{rm@Cnt}%
```

We take the 4th parameter, a named path, and pass it to `\useNamedPath`, then use `\filename@parse` to extract the components of the path.

```
  \rma@edefexecute{\noexpand\filename@parse{\rma@useNamedPath{#4}}}%
```

We record this named path as `rmFileStrm#4`, if this command is undefined, we define it. It's value is the indirect reference to the steam; if already defined, the command should contain the indirect reference to the same file that has been already embedded. If undefined we set a boolean to embed this stream, otherwise, we don't embed.

```
  \edef\rma@fs@expand{rmFileStrm#4}\@ifundefined{\rma@fs@expand}{%
  \rm@csarg\xdef{\rma@fs@expand}%
    {rmfstream\therm@Cnt}\global\rma@EmbedFiletrue}%
    {\global\rma@EmbedFilefalse}%
```

After having parsed the path, we now save the pieces for later use.

```
\edef\rma@thisfilepath{\filename@area}%
\edef\rma@basefilename{\filename@base}%
\edef\rma@extension{\filename@ext}%
\rma@edefexecute{\noexpand\uppercase{\noexpand
  \def\noexpand\rma@tempi{\rma@extension}}}%
```

Take a look at the file extension, if it is a 3D type extension, mark it as a RM3D annot by setting `\rma@isiiidtrue`.

```
\ifx\rma@tempi\rma@rmAnnot@type@uiiid\rma@isiiidtrue
  \else\ifx\rma@tempi\rma@rmAnnot@type@prc\rma@isiiidtrue
  \else\rma@isiiidfalse
\fi\fi
```

Process Options. We finally get around to processing the options. We put `\setkeys` in an `\edef` to allow the user to use macros to specify some of the options. The next line are the options passed by `#1` of `\rmAnnot`.

```
\rma@edefexecute{\noexpand\setkeys{rmAnnot}{#1}}%
```

If use has specified either the `width` or `height`, we reset the width and height of the annotation, while preserving the aspect ratio. We first test `width` then `height`. If both are specified, we use the `width` key and ignore the `height` key.

```
\bgroup
\ifx\rmAnnot@width\@empty
  \ifx\rmAnnot@height\@empty
```

If `width` and `height` are not specified, we determine if the `scale` key is listed.

```
    \ifx\rmAnnot@scale\@empty\else
      \setlength{\dimen@}%
        {\rm@Annot@height*\real{\rmAnnot@scale}}%
      \xdef\rm@Annot@height{\the\dimen@}%
      \setlength{\dimen@}%
        {\rm@Annot@width*\real{\rmAnnot@scale}}%
      \xdef\rm@Annot@width{\the\dimen@}%
    \fi
  \else
```

If `height` but not `width` is specified, we resize accordingly.

```
    \setlength{\dimen@}%
      {\rmAnnot@height*\ratio
        {\rm@Annot@width}{\rm@Annot@height}}%
    \xdef\rm@Annot@width{\the\dimen@}%
    \setlength{\dimen@}{\rmAnnot@height}%
    \xdef\rm@Annot@height{\the\dimen@}%
  \fi
\else
```

If `width` but not `height` is specified, we resize accordingly.

```
  \setlength{\dimen@}%
    {\rmAnnot@width*\ratio
      {\rm@Annot@height}{\rm@Annot@width}}%
```

```
   \xdef\rm@Annot@height{\the\dimen@}%
   \setlength{\dimen@}{\rmAnnot@width}%
   \xdef\rm@Annot@width{\the\dimen@}%
 \fi
 \egroup
\ifx\rma@input@iiidCode\relax\else
\ifrma@isiiid
```

Process the options of the RM3D annot, as passed to us by `\setRmOptions3D`.

```
 \@ifundefined{\rma@Annot@name_3DOPTS}{%
```

To Do. Here, we can insert some default options if the user did not specify anything. Including a reminder to make a declaration.

```
 }{%
```

If the user has specified the `use3D` option and the file specified in `#4` is a 3D model (U3D or PRC), then the switch `\ifrma@isiiid` is true, and we process any 3D options as specified by `\setRmOptions3D`. The command sequence `\rma@Annot@name_3DOPTS` is defined for this annot by `\setRmOptions3D`; we only process if there are options for this annot.

```
   \def\rma@Instances{}\def\rma@appendToNameTree{}%
```

Expand the arguments of `\setkeys` before allowing `\setkeys` to execute.

```
   \rma@edefexecute{\noexpand\setkeys{rm3DOptsTopLevel}%
     {\@nameuse{\rma@Annot@name_3DOPTS}}}%
   \edef\additional@Instances{\rma@Instances}%
   \edef\rma@addResources{\rma@appendToNameTree}%
```

3Djscript: We determine if there is one or more javascript files specified.

```
   \ifx\rma@rmAnnot@iiiDjs\@empty\else
     \def\@MXV@jscriptiiid{}%
     \literalps@out{%
       \ps@mark/_objdef {jscriptiiid\therm@Cnt}%
         /type/array/OBJ pdfmark^^J%
       \rmiiid@addToScriptsArray
       \rma@addFileSpecs
     }%
```

We save the key-value pair for the `Scripts` key.

```
     \edef\@MXV@jscriptiiid{%
         /Scripts {jscriptiiid\therm@Cnt}%
     }%
   \fi
```

Build the array of 3D views

```
   \@MXV@buildva%
 }%
\fi\fi
```

Identify the Extension. We try to identify the extension provided by the author. We take the `\filename@base` (saved as `\rma@extension`), convert it to upper case and compare with the text macros containing SWF, FLV, MP3, or one of the other supported extensions. The macro \rma@tempi contains the upper case form of the extension.

```
  \rma@edefexecute{\noexpand\uppercase{\noexpand
    \def\noexpand\rma@tempi{\rma@extension}}}%
```

3D Model. Search for a U3D or PRC. We earlier did a test to see if this is 3D or not, so the boolean \ifrma@isiiid has already been set.

```
\ifrma@isiiid
  \def\rma@poster@descrip{3D}%
  \def\rma@RMCSubtype{/3D}%
  \ifx\rma@tempi\rma@rmAnnot@type@uiiid
    \def\rma@rmAnnot@type{U3D}%
    \edef\rma@mimeType{\rma@mimetype@uiiid}%
  \else
  \ifx\rma@tempi\rma@rmAnnot@type@prc
    \def\rma@rmAnnot@type{PRC}%
    \edef\rma@mimeType{\rma@mimetype@prc}%
  \fi\fi
\else
```

Not a 3D model, so we'll check for more conventional types.

Flash Application. Search for a SWF file, we set the identifiers for later use.

```
  \def\rma@poster@descrip{Flash}%
  \ifx\rma@tempi\rma@rmAnnot@type@swf
    \def\rma@rmAnnot@type{SWF}\edef\rma@mimeType{\rma@mimetype@swf}%
    \def\rma@RMCSubtype{/Flash}%
  \else % if not flash
```

Video Formats. We search for extension that is generally classified as video.

```
  \def\rma@poster@descrip{Video}\def\rma@rmAnnot@type{FLV}%
  \@FndSuppExtfalse
  \@tfor\rma@type:={flv}{fiv}{mpiv}{mivV}{mov}{iiiGP}{iiiGii}\do{%
    \expandafter\ifx\expandafter\rma@tempi\csname%
     rma@rmAnnot@type@\rma@type\endcsname
     \@FndSuppExttrue
      \edef\rma@mimeType{\csname%
        rma@mimetype@\rma@type\endcsname}%
      \rma@recordVideoPlayer\@break@tfor
    \fi
  }%
  \if@FndSuppExt
    \def\rma@RMCSubtype{/Video}%
    \xdef\FileStrmVideoPlayer{rmVideoPlayer\therm@Cnt}%
  \else % if not video
```

Audio Formats. Test for a MP3 file, we need to embed the AudioPlayer once and only once. The Boolean \ifrma@EmbedAudioPlayer and the marker \rma@isAPEmbedded are used to keep track of whether the player has been embedded.

```
  \ifx\rma@tempi\rma@rmAnnot@type@mpiii
    \def\rma@rmAnnot@type{MP3}\edef\rma@mimeType{\rma@mimetype@mpiii}%
    \def\rma@poster@descrip{MP3}\def\rma@RMCSubtype{/Sound}%
    \xdef\FileStrmAudioPlayer{rmAudioPlayer\therm@Cnt}%
    \let\rma@rmAnnot@resources\@empty
    \rma@recordAudioPlayer
  \else % not mp3
```

Error. The extension is not recognized.

```
    \rma@PkEr@i
```

end testing for 3D (u3d and prc), flash, video, and audio

```
\fi\fi\fi\fi
```

Define \rma@thisfileName and \rma@fullpath for later use.

```
  \def\rma@thisfileName{\rma@basefilename.\rma@extension}%
  \def\rma@fullpath{\rma@thisfilepath\rma@thisfileName}%
```

If this is an FLV video file, we don't let the user created flash variables

```
  \ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
    \let\rma@rmAnnot@flashvars\@empty\fi
  \ifx\rma@rmAnnot@type\rma@rmAnnot@type@mpiii
    \let\rma@rmAnnot@flashvars\@empty\fi
```

If there is no poster, we supply one. For MP3, it is an image of the AudioPlayer controls, otherwise, it is the default poster.

```
\ifx\rma@rmAnnot@poster\@empty
  \ifKV@rmAnnot@defaultposter
    \Gin@defaultbp\this@width\rm@Annot@width
    \Gin@defaultbp\this@height\rm@Annot@height
    \ifdim\rm@Annot@width < \rm@Annot@height
      \edef\calc@prop{\this@width}\else
      \edef\calc@prop{\this@height}\fi
    \def\this@bbox{0 0 \this@width\space\this@height}%
    \begin{sp@createImage}{\this@bbox}{rmAP@#4@\therm@Cnt}%
    \rma@invisible
    \rma@psgraphics@poster
    \end{sp@createImage}%
    \def\rma@rmAnnot@poster{rmAP@#4@\therm@Cnt}%
\else
    \ifx\rma@rmAnnot@type\rma@rmAnnot@type@mpiii
      \def\rma@rmAnnot@poster{nramp3poster}%
    \else
        \Gin@defaultbp\this@width\rm@Annot@width
        \Gin@defaultbp\this@height\rm@Annot@height
        \ifdim\rm@Annot@width < \rm@Annot@height
```

```
          \edef\calc@prop{\this@width}\else
          \edef\calc@prop{\this@height}\fi
        \def\this@bbox{0 0 \this@width\space\this@height}%
        \begin{sp@createImage}{\this@bbox}{rmAP@#4@\therm@Cnt}%
        \rma@invisible
        \rma@psgraphics@poster
        \end{sp@createImage}%
        \def\rma@rmAnnot@poster{rmAP@#4@\therm@Cnt}%
    \fi
  \fi
\fi
```

Begin the construction of the RMA. Place the dimensions input by the author in a \Bbox within \pdf@rect. \Bbox is defined in the eforms package.

```
  \pdf@rect{\Bbox{\rm@Annot@width}{\rm@Annot@height}}%
```

```
  \@MXV@newlabel{rmAnnot_\rma@Annot@name}{rmAnnot\therm@Cnt}%
  \@MXV@labeltoaux{rmAnnot_\rma@Annot@name}{rmAnnot\therm@Cnt}%
```

Begin writing the rich media annotation through a PostScript special. The command \literalps@out is defined in hyperref.

```
  \literalps@out{%
```

Create the RichMedia Annotation.

```
    \ps@mark/_objdef {rmAnnot\therm@Cnt}%
      /Type/Annot%
      /Subtype/RichMedia%
      /NM (\rma@Annot@name)%                  % Annotation name
\ifx\rma@rmAnnot@poster\@empty\else
      /AP <</N {\rma@rmAnnot@poster}>>%       % poster appearance
\fi
      /F 68%                                  % Annotation flags
      /P {ThisPage}%                          % Parent
      /Border [ 0 0 \rma@rmAnnot@borderwidth ]% Border
      /BS <</Type/Border%                     % Border Style dictionary
        /W \rma@rmAnnot@borderwidth%        % Width
        /S/S%                               % Border style (Solid)
      >>%
```

The **RichMedia** annot has a **RichMediaContent** dictionary, and a **RichMediaSettings** dictionary, give indirect references to these.

```
    /RichMediaContent {rmContent\therm@Cnt}%
    /RichMediaSettings {rmSettings\therm@Cnt}
    H.B /ANN pdfmark^^J%
```

The RichMediaContent dictionary. We include the Configurations key, an array of indirect references to RichMediaConfiguration dictionaries, and the Assets key, an indirect reference to the Assets dictionary.

```
  \ps@mark/_objdef {rmContent\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmContent\therm@Cnt} <<%
    /Type/RichMediaContent%
```

If this is a RM3DA, we declare it in the RichMediaContent dictionary. Here we declare /Subtype/3D and insert a views array.

```
\ifrma@isiiid
    /Subtype/3D%
\ifx\@MXV@varray\@empty\else
    /Views [\@MXV@varray]%
\fi\fi
    /Configurations [{rmConfig\therm@Cnt}]%
    /Assets {rmAssets\therm@Cnt}%
  >>/PUT pdfmark^^J%
```

The RichMediaConfiguration dictionary. We set the primary content type of the configuration (Flash, Video, or Sound), and reference the corresponding instances, the SWF file to play, the VideoPlayer to play a FLV file, and the AudioPlayer, to play MP3 files.

```
    \ps@mark/_objdef {rmConfig\therm@Cnt}/type/dict/OBJ pdfmark^^J%
    \ps@mark{rmConfig\therm@Cnt} <<%
      /Type/RichMediaConfiguration%
      /Name (RMConfig\therm@Cnt)%
```

The Subtype is 3D, Flash, Video, or Sound, here, \rma@RMCSubtype was determined earlier.

```
      /Subtype\rma@RMCSubtype%
      /Instances {rmInstances\therm@Cnt}%
    >> /PUT pdfmark^^J%
```

The Instances Array.

This version of the Instances array is used, same as the SWF version.

```
    \ps@mark/_objdef {rmInstances\therm@Cnt}/type/array/OBJ pdfmark^^J%
\ifrma@isiiid
    \ps@mark{rmInstances\therm@Cnt} {rmInstance\therm@Cnt}%
      /APPEND pdfmark^^J%
    \additional@Instances
\else
```

We load the indirect reference to the VideoPlayer

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
    \ps@mark{rmInstances\therm@Cnt} {rmVideoPlayer\therm@Cnt}%
      /APPEND pdfmark^^J%
\else\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf
```

The Instances array will be populated later by resources.

```
    \ps@mark{rmInstances\therm@Cnt} {rmInstance\therm@Cnt}%
      /APPEND pdfmark^^J%
\else
```

We load the indirect reference to the AudioPlayer

```
    \ps@mark{rmInstances\therm@Cnt} {rmAudioPlayer\therm@Cnt}%
      /APPEND pdfmark^^J%
\fi\fi\fi
```

The Assets name tree. Reference by the `RichMediaContent` dictionary, this name tree lists the files needed, and indirect references to their respective file specification dictionaries. The assets are a function of whether the file type is SWF, FLV or MP3. For FLV, there are a number of skins that can be selected from, so the skin select through the skin key is included as assets, and its corresponding SWF needs to be embedded, if not embedded already.

```
  \ps@mark/_objdef {rmAssets\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmAssets\therm@Cnt} <<%
    /Names {\rma@ANT}>>/PUT pdfmark^^J%
  \ps@mark/_objdef {\rma@ANT}/type/array/OBJ pdfmark^^J%
\ifrma@isiiid
  \ps@mark{\rma@ANT} (\rma@thisfileName) /APPEND pdfmark^^J%
  \ps@mark{\rma@ANT} {rmfilespec\therm@Cnt} /APPEND pdfmark^^J%
  \rma@addResources
\else
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf
  \ifKV@rmAnnot@url\else
    \ps@mark{\rma@ANT} (\rma@thisfileName)/APPEND pdfmark^^J%
    \ps@mark{\rma@ANT} {rmfilespec\therm@Cnt}/APPEND pdfmark^^J%
    \rma@addResources
  \fi
\else\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
    \ifKV@rmAnnot@url\else
      \ps@mark{\rma@ANT} (\rma@thisfileName)/APPEND pdfmark^^J%
      \ps@mark{\rma@ANT} {rmfilespec\therm@Cnt}/APPEND pdfmark^^J%
      \ifVideoPlayerEx\rma@addResources\fi
    \fi
    \ifx\rma@rmAnnot@Skin\@empty\else
      \ps@mark{\rma@ANT} (\rma@rmAnnot@Skin)/APPEND pdfmark^^J%
      \ps@mark{\rma@ANT} {rmfilespecSkin\rma@skinName}%
        /APPEND pdfmark^^J%
    \fi
    \ps@mark{\rma@ANT} (\rma@VideoPlayer)/APPEND pdfmark^^J%
    \ps@mark{\rma@ANT} {rmfilespecVP}/APPEND pdfmark^^J%
\else
    \ps@mark{\rma@ANT} (AudioPlayer.swf)/APPEND pdfmark^^J%
    \ps@mark{\rma@ANT} {rmfilespecAP}/APPEND pdfmark^^J%
    \ps@mark{\rma@ANT} (\rma@thisfileName)/APPEND pdfmark^^J%
    \ps@mark{\rma@ANT} {rmfilespec\therm@Cnt}/APPEND pdfmark^^J%
\fi\fi\fi
```

The `RichMediaInstance` dictionary. Describes a single instance of an asset. The asset being described is the `VideoPlayer` (in the case of FLV files), the Flash file in the case of SWF files, and the `AudioPlayer` (for MP3 files).

```
\ifrma@isiiid
  \ps@mark/_objdef {rmInstance\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmInstance\therm@Cnt}%
\else
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
```

```
  \ps@mark/_objdef {rmVideoPlayer\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmVideoPlayer\therm@Cnt}%
\else\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf
  \ps@mark/_objdef {rmInstance\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmInstance\therm@Cnt}%
\else
  \ps@mark/_objdef {rmAudioPlayer\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmAudioPlayer\therm@Cnt}%
\fi\fi\fi
    <<%
      /Type/RichMediaInstance%
\ifrma@isiiid
      /Subtype/3D%
      /Asset {rmfilespec\therm@Cnt}%
\else
      /Subtype/Flash%
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
      /Asset {rmfilespecVP}%
\else\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf
      /Asset {rmfilespec\therm@Cnt}%
\else
      /Asset {rmfilespecAP}%
\fi\fi\fi
\ifrma@isiiid\else
      /Params {rmParams\therm@Cnt}%
\fi
    >> /PUT pdfmark^^J%
```

The Params dictionary. Contains parameters related to an active Flash subtype in a RichMediaInstance dictionary; the user, through the UI or through this package, can set the value of the FlashVars entry. For content subtypes other than Flash, the argument for FlashVars is not under user control, and is determined by the PDF creation application, this package.

```
  \ps@mark/_objdef {rmParams\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmParams\therm@Cnt} <<%
    /Type/RichMediaParams%
```

If there are no additional resources specified, we bind the animation to the background, if there are (SWF) resources, we bind to the foreground.

```
\ifrma@isiiid\else
\ifx\rma@rmAnnot@resources\@empty
    /Binding/Background%
\else
    /Binding/Foreground%
\fi\fi
```

If this is a FLV (video), we use the custom flash variables of Acrobat (reverse engineering). If it is a url, we specify the full URL, otherwise, we specify the file name as the source.

```
\ifrma@isiiid\else
```

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
  \ifKV@rmAnnot@url
    /FlashVars (source=\rma@fullpath&%
  \else
    /FlashVars (source=\rma@thisfileName&%
  \fi
  \ifx\rma@rmAnnot@Skin\@empty\else
    skin=\rma@rmAnnot@Skin&%
  \fi
    skinAutoHide=\rma@skinAutoHide&%
    skinBackgroundColor=\rma@skinBGColor&%
    skinBackgroundAlpha=\rma@skinBGAlpha&%
    volume=\rma@rmAnnot@volume)
\ifx\rma@rmAnnot@cuepoints\@empty\else
    /CuePoints [\rma@array@hold]%
\fi
\else\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf
```

If this is a SWF file, we allow the author to introduce custom flash variables, hope s/he knows what is s/he is doing.

```
    \ifx\rma@rmAnnot@flashvars\@empty\else
      /FlashVars (\rma@rmAnnot@flashvars)%
    \fi
\else
      /FlashVars (source=\ifKV@rmAnnot@url\rma@fullpath\else
        \rma@thisfileName\fi&autoPlay=true&%
        volume=\rma@rmAnnot@volume)%
\fi\fi\fi
    >> /PUT pdfmark^^J%
```

The RichMediaSettings Dictionary. The second dictionary referenced in the RichMedia annot is the RichMediaSettings dictionary. The RichMediaSettings dictionary stores the conditions and responses that occur in response to certain events, such as activation and deactivation of the annotation, and contains two dictionaries.

```
  \ps@mark/_objdef {rmSettings\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmSettings\therm@Cnt} <<%
    /Type/RichMediaSettings%
```

The Activation key is a dictionary that describes how the annot is to be activated and played.

```
    /Activation <<%
      /Type/RichMediaActivation%
      /Condition\rma@rmAnnot@enabled
      /Configuration {rmConfig\therm@Cnt}%
```

(2011/11/08) Used for Keyframe animation, normally not needed. Will uncomment when I develop an example of usage. Note, the speed and playcount keys are now ignored.

```
\ifrma@isiiid
```

```
      /Animation%
        <<%
          /Type/RichMediaAnimation%
          /Subtype/Linear%
          /Speed \rma@rmAnnot@speed
          /PlayCount \rma@rmAnnot@playcount
        >>%
\ifx\@MXV@defaultview\@empty\else
      /View \@MXV@defaultview
\fi
\ifx\@MXV@jscriptiiid\@empty\else
      \@MXV@jscriptiiid
\fi\fi
      /Presentation {rmPresentation\therm@Cnt}%
    >>
```

The Deactivation key is a dictionary that describes how the annot is to be deactivated.

```
      /Deactivation<<%
        /Type/RichMediaDeactivation%
        /Condition\rma@rmAnnot@deactivated
      >>%
  >>/PUT pdfmark^^J%
```

The CuePoints Array.

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
\ifx\rma@rmAnnot@cuepoints\@empty\else\rma@dict@hold\fi\fi
```

The RichMediaPresentation Dictionary. Referenced within the Activation entry of the RichMediaSettings dictionary above, this dictionary determines whether the background is transparent, and whether the media is to be played within the page or in a floating window.

```
  \ps@mark/_objdef {rmPresentation\therm@Cnt}%
    /type/dict/OBJ pdfmark^^J%
  \ps@mark{rmPresentation\therm@Cnt}<<%
    /Type/RichMediaPresentation%
\ifrma@isiiid
    /NavigationPane \rma@rmAnnot@modeltree % need key
    /Toolbar \rma@rmAnnot@toolbar % need key
    /Transparent \rma@rmAnnot@transparent
\else
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf
    /Transparent \rma@rmAnnot@transparent
    /PassContextClick \rma@rmAnnot@PassContextClick
    /NavigationPan false%
\else
    /Transparent false%
    /NavigationPan false%
\fi\fi
\ifKV@rmAnnot@windowed
```

```
    /Style/Windowed%
    /Window {rmWindow\therm@Cnt}%
\else
    /Style/Embedded%
\fi
  >>/PUT pdfmark^^J%
```

The RichMediaWindow Dictionary. When the style is Windowed as specified in the `RichMediaPresentation` dictionary, we include the `RichMediaWindow` dictionary to set the parameters of the window. Currently, the parameters are hard-wired to the defaults, as specified by the *Adobe Suppl. Doc.*

```
\ifKV@rmAnnot@windowed
  \ps@mark/_objdef {rmWindow\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmWindow\therm@Cnt}<<%
    /Type/RichMediaWindow%
    /Height<<%
      /Default \rma@winDimPosHeight@def
      /Max \rma@winDimPosHeight@max
      /Min \rma@winDimPosHeight@min
    >>%
    /Width<<%
      /Default \rma@winDimPosWidth@def
      /Max \rma@winDimPosWidth@max
      /Min \rma@winDimPosWidth@min
    >>%
    /Position<<%
      /Type/RichMediaPosition % RichMediaPosition dictionary
      /HAlign\rma@winDimPosPos@halign
      /VAlign\rma@winDimPosPos@valign
      /HOffset \rma@winDimPosPos@hoffset
      /VOffset \rma@winDimPosPos@voffset
    >>%
  >>/PUT pdfmark^^J%
\fi
```

File Specifications and Streams. In this section, we insert the Filespec dictionaries of the various assets. There are four type of assets: (1) the one specified by the 4th argument of the `\rmAnnot`; (2) additional assets specified through the `resources` key; (3) the VideoPlayer; and (4) the AudioPlayer. The first two we'll call *Author Supplied Assets*, the latter two, we'll call *System Supplied Assets*.

Author Supplied Assets. We first insert the file spec and stream for the 4th parameter, its filename is `\rma@thisfileName` and its full path name is `\rma@fullpath`.

File specs: Filespec dictionary

```
  \ps@mark/_objdef {rmfilespec\therm@Cnt}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmfilespec\therm@Cnt} <<%
\ifKV@rmAnnot@url
```

```
    /F(\rma@fullpath)%
    /FS/URL%
\else
    /F(\rma@thisfileName)%
    /UF (\rma@thisfileName)%
    /EF <</F {\csname rmFileStrm#4\endcsname}>>
\fi
    /Type/Filespec
  >>/PUT pdfmark^^J%
```

File stream: EmbeddedFile dictionary

```
\ifKV@rmAnnot@url\else
\ifrma@EmbedFile
  \ps@mark/_objdef {\csname rmFileStrm#4\endcsname}%
    /type/stream/OBJ pdfmark^^J%
  \ps@mark{\csname rmFileStrm#4\endcsname} (\rma@fullpath)
    (r) file /PUT pdfmark^^J%
  \ps@mark{\csname rmFileStrm#4\endcsname} <<%
    /Type/EmbeddedFile%
    /Subtype(\rma@mimeType)%
  >>/PUT pdfmark^^J%
  \ps@mark{\csname rmFileStrm#4\endcsname} /CLOSE pdfmark^^J%
\fi\fi
```

If we are dealing with a SWF or 3D file, we'll then include additional file specs and streams as specified by the `resources` key of `\rmAnnot`.

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@swf\rma@addFileSpecs\fi
\ifrma@isiiid\rma@addFileSpecs\fi
```

System Supplied Assets. We have the various skins, the `VideoPlayer`, and the `AudioPlayer`. We'll start with the skins.

File specs for skin

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
\ifVideoPlayerEx\rma@addFileSpecs\fi
\ifx\rma@rmAnnot@Skin\@empty\else
  \ps@mark/_objdef {rmfilespecSkin\rma@skinName}%
    /type/dict/OBJ pdfmark^^J%
  \ps@mark{rmfilespecSkin\rma@skinName} <<%
    /F (\rma@rmAnnot@Skin)%
    /Type/Filespec%
    /UF (\rma@rmAnnot@Skin)%
    /EF <</F {rmfstreamSkin\rma@skinName}>>
  >>/PUT pdfmark^^J%
```

File stream for skin

```
\rm@csarg\if{embedSkin\rma@skinName}\rm@One
  \ps@mark/_objdef {rmfstreamSkin\rma@skinName}%
    /type/stream/OBJ pdfmark^^J%
  \ps@mark{rmfstreamSkin\rma@skinName}%
    (\rma@pathToSkins/\rma@rmAnnot@Skin) (r) file%
    /PUT pdfmark^^J%
  \ps@mark{rmfstreamSkin\rma@skinName} <<%
    /Type/EmbeddedFile
    /Subtype (\rma@mimetype@swf)
  >>/PUT pdfmark^^J%
  \ps@mark{rmfstreamSkin\rma@skinName}/CLOSE pdfmark^^J%
\fi\fi\fi
```

Now the specs and stream of the `VideoPlayer`

File specs for video player

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@flv
  \ps@mark/_objdef {rmfilespecVP}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmfilespecVP} <<%
    /Type/Filespec%
    /F (\rma@VideoPlayer)%
    /UF (\rma@VideoPlayer)%
    /EF <</F {rmfstreamVP}>>
  >>/PUT pdfmark^^J%
```

We'll only embed once, provided `\ifrma@EmbedVideoPlayer` is true.

File stream for video player

```
\ifrma@EmbedVideoPlayer
  \ps@mark/_objdef {rmfstreamVP}/type/stream/OBJ pdfmark^^J%
  \ps@mark{rmfstreamVP} (\rma@pathToPlayers/\rma@VideoPlayer)
    (r) file /PUT pdfmark^^J%
  \ps@mark{rmfstreamVP} <<%
    /Type/EmbeddedFile%
    /Subtype (\rma@mimetype@swf)%
  >>/PUT pdfmark^^J%
  \ps@mark{rmfstreamVP} /CLOSE pdfmark^^J%
\fi\fi
```

Now the specs and stream of the `AudioPlayer`

File specs for audio player

```
\ifx\rma@rmAnnot@type\rma@rmAnnot@type@mpiii
  \ps@mark/_objdef {rmfilespecAP}/type/dict/OBJ pdfmark^^J%
  \ps@mark{rmfilespecAP} <<%
    /F (AudioPlayer.swf)%
    /Type/Filespec%
    /UF (AudioPlayer.swf)%
    /EF <</F {rmfstreamAP}>>%
  >>/PUT pdfmark^^J%
```

We'll only embed once, provided `\ifrma@EmbedAudioPlayer` is true.

File stream for audio player

```
\ifrma@EmbedAudioPlayer
  \ps@mark/_objdef {rmfstreamAP}/type/stream/OBJ pdfmark^^J%
  \ps@mark{rmfstreamAP}(\rma@pathToPlayers/AudioPlayer.swf)
    (r) file /PUT pdfmark^^J%
  \ps@mark{rmfstreamAP} <<%
    /Type/EmbeddedFile%
    /Subtype (\rma@mimetype@mpiii)%
  >>/PUT pdfmark^^J%
  \ps@mark{rmfstreamAP} /CLOSE pdfmark^^J%
\fi\fi}%
```

If we are using user defined skins, we set the skin just used so that this skin will not be embedded a second time, let's hope.

```
\ifx\rma@rmAnnot@type@flv\rma@rmAnnot@type
\ifx\rma@rmAnnot@Skin\@empty\else
\expandafter\global\rm@csarg\let{embedSkin\rma@skinName}\rm@Zero
\fi\fi
```

End of `\rmAnnot`. Close off the group, and end the `\rmAnnot` command definition.

```
\endgroup}
```

4.3 Poster Appearances

`\defaultPoster` The command `\defaultPoster` defines the default poster appearance and is used when a poster appearance is not specified by the `poster` key. If the file is an MP3, then there is a graphic that is the default poster appearance.

`\defaultPoster` can be defined to something else, if desired. The commands should be PostScript graphic operators.

This command defines a text macro `\rma@psgraphics@poster` that is expanded in the definition of `\rmAnnot`, before the start of the annot itself.

```
\newcommand{\defaultPoster}[1]{\def\rma@psgraphics@poster{#1}}
\defaultPoster
{%
  \rma@ps@bg@setcolor
  0 0 \this@width\space\this@height\space rectfill
  \rma@ps@txt@x\adj@measure\rma@ps@txt@y\adj@measure moveto
  \rma@ps@txt@setcolor/\rma@ps@font
  \rma@ps@relfontsize\rma@ps@fontsize selectfont
  \rma@ps@msg
}
```

The definitions of the text macros that enable the document author to make minor changes in color, font, and placement of text, to the default poster.

```
\def\adj@measure{\calc@prop\space mul 100 div }
\def\rma@ps@bg@setcolor{.7529 setgray }
\def\rma@ps@txt@x{10 }\def\rma@ps@txt@y{10 }
\def\rma@ps@txt@setcolor{.4 setgray }
\def\rma@ps@font{Helvetica }
\def\rma@ps@relfontsize{10 \adj@measure}
\let\rma@ps@fontsize\@empty
\def\rma@ps@msg{(\rma@posternote) show}
```

`\setPosterProps{`⟨*KV-pairs*⟩`}` A convenience command to execute ⟨*KV-pairs*⟩ from the `rmPoster` family. Below are the key-values of the rmPoster family for designing your own custom default poster. All values are expressed in PostScript operators.

```
\define@key{rmPoster}{color}[]{\def\rma@ps@bg@setcolor{#1 }}
\define@key{rmPoster}{xPos}[]{\def\rma@ps@txt@x{#1 }}
\define@key{rmPoster}{yPos}[]{\def\rma@ps@txt@y{#1 }}
\define@key{rmPoster}{textColor}[]{\def\rma@ps@txt@setcolor{#1 }}
\define@key{rmPoster}{relTextSize}[]{\def
    \rma@ps@relfontsize{#1 \adj@measure}%
    \let\rma@ps@fontsize\@empty}
\define@key{rmPoster}{textSize}[]{\def\rma@ps@fontsize{#1 }%
    \let\rma@ps@relfontsize\@empty}
\define@key{rmPoster}{textFont}[]{\def\rma@ps@font{#1 }}
\def\setPosterProps#1{\setkeys{rmPoster}{#1}}
```

\makePoster A convenience command for making posters. Assuming you have an eps of the appropriate aspect ratio to use as poster, you can say

\makePoster[⟨*graphics_options*⟩]{⟨*name*⟩}{⟨*graphics_file*⟩}

for example

```
\makePoster[hiresbb]{AcroAd_poster}{AcroAd_poster}
```

Then you can say

```
\rmAnnot[poster=AcroAd_poster]{612bp}{265bp}{AcroAd}
```

```
\providecommand{\makePoster}[3][]{%
  \embedEPS[#1]{rma@#2}{#3}%
  \begin{createImage}{\bboxOf{rma@#2}}{#2}%
      \ps@mark{rma@#2} /SP pdfmark
  \end{createImage}%
}
\@onlypreamble{\makePoster}
```

Finally, we define several error messages.

```
\def\rma@PkEr@i{%
  \PackageError{rmannot}{%
    You must specify a file with an extension\MessageBreak
    of .swf, .flv, .f4v, .mp4, .m4v, .mov, .3gp,\MessageBreak
    .3g2, .mp3}{Specify one of the supported file extensions to
    embed in this annotation.\MessageBreak
    See the rmannot manual for details on supported extensions.}}
\def\rma@PkEr@ii{%
  \PackageError{rmannot}{%
    The name `\rm@argii' has already been used. Either\MessageBreak
    you are defining the same path, or a different path\MessageBreak
    with the same name}{%
    Names must be unique to the document, choose another}}
\def\rma@PkEr@iii#1{%
  \PackageError{rmannot}{%
```

```
    No extension supplied with this file name,\MessageBreak#1.%
    \MessageBreak Please include a file extension of\MessageBreak
     .swf, .flv, or .mp3, as appropriate}{%
    Include an extension of .swf, .flv, or .mp3}}
⟨/package⟩
```

4.4 Support for 3D Annotations

This section supports of options of the command `\setRmOptions3D`, used for passing a set of options to a RM3DA. At this writing, there are two recognized keys, `3DResources` and `3DOptions`. The former holds the key-values of my own construction; the latter holds the key-values of the `movie15` package by Alexander Grahn.

```
⟨*3Dcode⟩
```

```
\setRmOptions3D{myDice}
{
    3DOptions={⟨options-from-movie15⟩},
    3DResources={
       none={rName=⟨name1⟩},...,
       foreground={rName=⟨name2⟩,flashvars=⟨vars⟩},...,
       background={rName=⟨name3⟩,flashvars=⟨vars⟩},...,
       material={rName=⟨name4⟩,mName=<materialName>,flashvars=⟨vars⟩},...
   }
}
```

3DOptions (3D)
3DResources (3D)

The `rm3DOptsTopLevel` family supports to top-level keys of `\setRmOptions3D`: `3DOptions` and `3DResources`.

```
\define@key{rm3DOptsTopLevel}{3DOptions}{%
  \def\rmiiiDTLOpts{#1}%
```

This `\define@key`, in turn, executes the family `MXV@user`, taken from movie15.

```
  \setkeys{MXV@user}{#1}%
}
\define@key{rm3DOptsTopLevel}{3DResources}{%
  \def\rmiiiDOptsTLRes{#1}%
```

This `\define@key`, in turn, executes the family `rm3DOpts`, an original family defined in this package.

```
  \setkeys{rm3DOpts}{#1}%
}
```

3DResources. This key recognizes the keys `none`, `foreground`, `background`, and `material`. The keys are optional, and may be repeated more than once. This paragraph processes the keys of the `rm3DOpts` family.

`\rma@ckFileForEmbed` determines if the current resource is already embedded, and defines `\rma@embed` to indicated this, if not embedded, it defines the `rmFileStrm#1`, and defines `\rm@irfstrm` for later use.

```
\def\rma@ckFileForEmbed#1#2{%
  \edef\rma@fs@expand{rmFileStrm#1}%
  \@ifundefined{\rma@fs@expand}{%
    \rm@csarg\xdef{\rma@fs@expand}{%
      rmfstream\therm@Cnt-#2#1}\def\rma@embed{1}}%
      {\def\rma@embed{0}}%
  \edef\rm@irfstrm{\@nameuse{rmFileStrm#1}}%
}
```

none (3D)

The none key

```
\define@key{rm3DOpts}{none}{%
```

We use nested key-values, the none key calls the family rm3DOpts@no, defined further below.

```
  \setkeys{rm3DOpts@no}{rName,#1}%
```

Parse the symbolic name, by using \filename@parse.

```
  \rma@edefexecute{\noexpand
    \filename@parse{\rma@useNamedPath{\rmiiiDOpts@no@rName}}}%
  \rma@ckFileForEmbed{\rmiiiDOpts@no@rName}{NONE}%
```

Add to the instance array, file specs/stream, and name tree.

```
  \edef\rma@Instances{\rma@Instances
      \ps@mark{rmInstances\therm@Cnt} %
      {rmInstance\therm@Cnt_NONE\rmiiiDOpts@no@rName}%
      /APPEND pdfmark^^J%
      \ps@mark/_objdef{rmInstance\therm@Cnt_NONE%
            \rmiiiDOpts@no@rName}/type/dict/OBJ pdfmark^^J%
      \ps@mark{rmInstance\therm@Cnt_NONE\rmiiiDOpts@no@rName}<<%
        /Asset {rmfilespec\therm@Cnt-NONE\rmiiiDOpts@no@rName}%
        /Type/RichMediaInstance>>/PUT pdfmark^^J%
      \rm@appendFileSpecs{NONE\rmiiiDOpts@no@rName}%
        {\filename@area}{\filename@base.\filename@ext}%
        {\rma@embed}{\rm@irfstrm}{}%
  }%
  \edef\rma@appendToNameTree{\rma@appendToNameTree
    \rm@appendNameTree{NONE\rmiiiDOpts@no@rName}%
      {\filename@area}{\filename@base.\filename@ext}%
      {\rma@embed}{\rm@irfstrm}{}%
  }%
}
```

foreground (3D)

The foreground key

```
\define@key{rm3DOpts}{foreground}{%
  \setkeys{rm3DOpts@fg}{rName,flashvars,#1}%
```

Parse the symbolic name, by using \filename@parse.

```
  \rma@edefexecute{\noexpand
    \filename@parse{\rma@useNamedPath{\rmiiiDOpts@fg@rName}}}%
  \rma@ckFileForEmbed{\rmiiiDOpts@fg@rName}{FG}%
```

Add to the instance array, file specs/stream, and name tree.

```
  \edef\rma@Instances{\rma@Instances
    \ps@mark{rmInstances\therm@Cnt} %
    {rmInstance\therm@Cnt_FG\rmiiiDOpts@fg@rName}%
    /APPEND pdfmark^^J%
    \ps@mark/_objdef{rmInstance\therm@Cnt_FG%
      \rmiiiDOpts@fg@rName}/type/dict/OBJ pdfmark^^J%
    \ps@mark{rmInstance\therm@Cnt_FG\rmiiiDOpts@fg@rName}<<%
      /Asset {rmfilespec\therm@Cnt-FG\rmiiiDOpts@fg@rName}%
      /Params <</Binding/Foreground%
        /FlashVars(\rmiiiDOpts@fg@flashvars)>>%
      /Type/RichMediaInstance%
    >>/PUT pdfmark^^J%
    \rm@appendFileSpecs{FG\rmiiiDOpts@fg@rName}{\filename@area}%
      {\filename@base.\filename@ext}{\rma@embed}%
      {\rm@irfstrm}{}%
  }%
  \edef\rma@appendToNameTree{\rma@appendToNameTree
    \rm@appendNameTree{FG\rmiiiDOpts@fg@rName}{\filename@area}%
      {\filename@base.\filename@ext}{\rma@embed}%
      {\rm@irfstrm}{}%
  }%
}
```

background (3D)

The background key

```
\define@key{rm3DOpts}{background}{%
  \setkeys{rm3DOpts@bg}{rName,flashvars,#1}%
```

Parse the symbolic name, by using \filename@parse.

```
  \rma@edefexecute{\noexpand
    \filename@parse{\rma@useNamedPath{\rmiiiDOpts@bg@rName}}}%
  \rma@ckFileForEmbed{\rmiiiDOpts@bg@rName}{BG}%
```

Add to the instance array, file specs/stream, and name tree.

```
  \edef\rma@Instances{\rma@Instances
    \ps@mark{rmInstances\therm@Cnt} %
    {rmInstance\therm@Cnt_BG\rmiiiDOpts@bg@rName}%
    /APPEND pdfmark^^J%
    \ps@mark/_objdef{rmInstance\therm@Cnt_BG%
      \rmiiiDOpts@bg@rName}/type/dict/OBJ pdfmark^^J%
    \ps@mark{rmInstance\therm@Cnt_BG\rmiiiDOpts@bg@rName}<<%
      /Asset {rmfilespec\therm@Cnt-BG\rmiiiDOpts@bg@rName}%
      /Params <</Binding/Background%
        /FlashVars(\rmiiiDOpts@bg@flashvars)>>%
        /Type/RichMediaInstance%
    >>/PUT pdfmark^^J%
    \rm@appendFileSpecs{BG\rmiiiDOpts@bg@rName}{\filename@area}%
      {\filename@base.\filename@ext}{\rma@embed}%
      {\rm@irfstrm}{}%
  }%
  \edef\rma@appendToNameTree{\rma@appendToNameTree
    \rm@appendNameTree{BG\rmiiiDOpts@bg@rName}{\filename@area}%
      {\filename@base.\filename@ext}{\rma@embed}%
      {\rm@irfstrm}{}%
  }%
}
```

material (3D) This key binds a resource to a material. The resource name is rName (as defined by \saveNamedPath), the key mName is the name of the material the resource is to be bound to; flashvars is used to pass variables to the SWF application.

```
\define@key{rm3DOpts}{material}{%
  \setkeys{rm3DOpts@mat}{rName,mName,flashvars,#1}%
```

Parse the symbolic name, by using \filename@parse.

```
  \rma@edefexecute{\noexpand
    \filename@parse{\rma@useNamedPath{\rmiiiDOpts@mat@rName}}}%
  \rma@ckFileForEmbed{\rmiiiDOpts@mat@rName}{MAT}%
```

Add to the instance array, file specs/stream, and name tree.

```
  \edef\rma@Instances{\rma@Instances
    \ps@mark{rmInstances\therm@Cnt} %
    {rmInstance\therm@Cnt_MAT\rmiiiDOpts@mat@rName}%
    /APPEND pdfmark^^J%
    \ps@mark/_objdef{rmInstance\therm@Cnt_MAT%
      \rmiiiDOpts@mat@rName}/type/dict/OBJ pdfmark^^J%
    \ps@mark{rmInstance\therm@Cnt_MAT\rmiiiDOpts@mat@rName}<<%
      /Asset {rmfilespec\therm@Cnt-MAT\rmiiiDOpts@mat@rName}%
      /Params <</Binding/Material%
        /BindingMaterialName(\rmiiiDOpts@mat@mName)%
        /FlashVars(\rmiiiDOpts@mat@flashvars)>>%
      /Type/RichMediaInstance%
    >>/PUT pdfmark^^J%
    \rm@appendFileSpecs{MAT\rmiiiDOpts@mat@rName}{\filename@area}%
      {\filename@base.\filename@ext}{\rma@embed}%
      {\rm@irfstrm}{}%
  }%
  \edef\rma@appendToNameTree{\rma@appendToNameTree
    \rm@appendNameTree{MAT\rmiiiDOpts@mat@rName}{\filename@area}%
      {\filename@base.\filename@ext}{\rma@embed}%
      {\rm@irfstrm}{}%
  }%
}
```

Process none values

```
\define@key{rm3DOpts@no}{rName}[]{\def\rmiiiDOpts@no@rName{#1}}
```

Process foreground values

```
\define@key{rm3DOpts@fg}{rName}[]{\def\rmiiiDOpts@fg@rName{#1}}
\define@key{rm3DOpts@fg}{flashvars}[]{\def\rmiiiDOpts@fg@flashvars{#1}}
```

Process background values

```
\define@key{rm3DOpts@bg}{rName}[]{\def\rmiiiDOpts@bg@rName{#1}}
\define@key{rm3DOpts@bg}{flashvars}[]{\def\rmiiiDOpts@bg@flashvars{#1}}
```

Process material values

```
\define@key{rm3DOpts@mat}{rName}[]{\def\rmiiiDOpts@mat@rName{#1}}
\define@key{rm3DOpts@mat}{mName}[]{\def\rmiiiDOpts@mat@mName{#1}}
\define@key{rm3DOpts@mat}{flashvars}[]{%
  \def\rmiiiDOpts@mat@flashvars{#1}}
```

4.4.1 Code from movie15

My gracious thanks to Alexander Grahn for granting permission to use some of his movie15 code.

```
\newread\@MXV@@viewsfile% file handle for views file
\newboolean{@MXV@eof}%
\newcount\@MXV@viewscount%counter for number of 3D views per inclusion
\newboolean{@MXV@viewsprovided}%3d views file provided?
\newboolean{@MXV@defaultviewprovided}%default 3D view provided?
\newcount\@MXV@nodecount% number of node dicts
\newcount\@MXV@cscount% number of cross section dicts
```

Default values

```
\def\@MXV@aac{30}% aperture angle of camera
\def\@MXV@roll{0}% camera roll angle
\def\@MXV@defaultbg{1 1 1}%
\def\@MXV@background{/BG<</CS/DeviceRGB/C[1 1 1]>>}%
\def\@MXV@defaultlights{}%
\def\@MXV@lights{}%
\def\@MXV@defaultrender{Solid}%
\def\@MXV@render{/RM <</Subtype/Solid>>}%
\def\@MXV@naentry{}% %takes array of Node dicts
\def\@MXV@saentry{}% %takes array of cross section  dicts
\let\@MXV@jscriptiiid\@empty
\let\rma@rmAnnot@iiiDjs\@empty
\let\@MXV@varray\@empty
\let\additional@Instances\@empty
\def\@MXV@defaultview{}
\def\@MXV@coo{0 0 0}% centre of orbit
\def\@MXV@ctoc{0 -1 0}% centre of orbit to camera vector
\def\@MXV@roo{0}% radius of orbit
\def\@MXV@viewsfileii{}%file of views of the 3D object (new format)
\setboolean{@MXV@viewsprovided}{false}%
\setboolean{@MXV@defaultviewprovided}{false}%
\def\@MXV@iiidview{}%
```

\@MXV@ciiwmatrix is a macro for building the transformation matrix

#1,#2,#3 centre of orbit coordinates (coo)

#4,#5,#6 centre of orbit to camera direction vector (c2c)

#7 orbital radius (roo)

#8 camera roll (roll)

```
\def\@MXV@ciiwmatrix#1 #2 #3 #4 #5 #6 #7 #8 {%
```

View vector (opposite to c2c)

```
  \FPupn\@MXV@viewx{#4 neg}%
  \FPupn\@MXV@viewy{#5 neg}%
  \FPupn\@MXV@viewz{#6 neg}%
```

Normalize view vector

```
  \FPupn\@MXV@modulo{\@MXV@viewx{} copy mul %
    \@MXV@viewy{} copy mul + %
    \@MXV@viewz{} copy mul + 2 swap root%
  }%
  \FPupn\@MXV@viewx{\@MXV@viewx{} \@MXV@modulo{} div}%
  \FPupn\@MXV@viewy{\@MXV@viewy{} \@MXV@modulo{} div}%
  \FPupn\@MXV@viewz{\@MXV@viewz{} \@MXV@modulo{} div}%
```

Camera roll

```
  \FPupn\@MXV@sinroll{#8 180.0 div \FPpi{} mul sin}%
  \FPupn\@MXV@cosroll{#8 180.0 div \FPpi{} mul cos}%
```

Top and bottom views

```
  \FPupn\@MXV@leftx{-1.0}%
  \FPupn\@MXV@lefty{0.0}%
  \FPupn\@MXV@leftz{0.0}%
  \FPifneg\@MXV@viewz% top view
    %up-vector
    \FPupn\@MXV@upx{0.0}%
    \FPupn\@MXV@upy{1.0}%
    \FPupn\@MXV@upz{0.0}%
  \else% bottom view
    %up-vector
    \FPupn\@MXV@upx{0.0}%
    \FPupn\@MXV@upy{-1.0}%
    \FPupn\@MXV@upz{0.0}%
  \fi%
  \FPupn\@MXV@sumxy{\@MXV@viewx{} abs \@MXV@viewy{} abs add}%
  \FPifeq\@MXV@sumxy{0}\else% other views than top and bottom
    %up-vector = up_world - (up_world dot view) view
    \FPupn\@MXV@upx{\@MXV@viewz{} \@MXV@viewx{} mul neg}%
    \FPupn\@MXV@upy{\@MXV@viewz{} \@MXV@viewy{} mul neg}%
    \FPupn\@MXV@upz{\@MXV@viewz{} \@MXV@viewz{} mul neg 1.0 add}%
    %normalize up-vector
    \FPupn\@MXV@modulo{\@MXV@upx{} copy mul \@MXV@upy{} copy %
      mul + \@MXV@upz{} copy mul + 2 swap root}%
    \FPupn\@MXV@upx{\@MXV@upx{} \@MXV@modulo{} div}%
    \FPupn\@MXV@upy{\@MXV@upy{} \@MXV@modulo{} div}%
    \FPupn\@MXV@upz{\@MXV@upz{} \@MXV@modulo{} div}%
    %left vector = up x view
```

```
 \FPupn\@MXV@leftx{\@MXV@viewz{} \@MXV@upy{} mul %
  \@MXV@viewy{} \@MXV@upz{} mul sub}%
 \FPupn\@MXV@lefty{\@MXV@viewx{} \@MXV@upz{} mul %
  \@MXV@viewz{} \@MXV@upx{} mul sub}%
 \FPupn\@MXV@leftz{\@MXV@viewy{} \@MXV@upx{} mul %
  \@MXV@viewx{} \@MXV@upy{} mul sub}%
 %normalize left vector
 \FPupn\@MXV@modulo{\@MXV@leftx{} copy mul \@MXV@lefty{} %
  copy mul + \@MXV@leftz{} copy mul + 2 swap root}%
 \FPupn\@MXV@leftx{\@MXV@leftx{} \@MXV@modulo{} div}%
 \FPupn\@MXV@lefty{\@MXV@lefty{} \@MXV@modulo{} div}%
 \FPupn\@MXV@leftz{\@MXV@leftz{} \@MXV@modulo{} div}%
\fi%
```

Apply camera roll

```
\FPupn\@MXV@leftxprime{\@MXV@leftx{} \@MXV@cosroll{} mul %
  \@MXV@upx{} \@MXV@sinroll{} mul +}%
\FPupn\@MXV@leftyprime{\@MXV@lefty{} \@MXV@cosroll{} mul %
  \@MXV@upy{} \@MXV@sinroll{} mul +}%
\FPupn\@MXV@leftzprime{\@MXV@leftz{} \@MXV@cosroll{} mul %
  \@MXV@upz{} \@MXV@sinroll{} mul +}%
\FPupn\@MXV@upxprime{\@MXV@upx{} \@MXV@cosroll{} mul %
  \@MXV@leftx{} \@MXV@sinroll{} mul sub}%
\FPupn\@MXV@upyprime{\@MXV@upy{} \@MXV@cosroll{} mul %
  \@MXV@lefty{} \@MXV@sinroll{} mul sub}%
\FPupn\@MXV@upzprime{\@MXV@upz{} \@MXV@cosroll{} mul %
  \@MXV@leftz{} \@MXV@sinroll{} mul sub}%
\FPupn\@MXV@leftx{\@MXV@leftxprime}%
\FPupn\@MXV@lefty{\@MXV@leftyprime}%
\FPupn\@MXV@leftz{\@MXV@leftzprime}%
\FPupn\@MXV@upx{\@MXV@upxprime}%
\FPupn\@MXV@upy{\@MXV@upyprime}%
\FPupn\@MXV@upz{\@MXV@upzprime}%
```

Translation vector

```
\FPupn\@MXV@roo{#7 abs}%
\FPifeq\@MXV@roo{0}\FPupn\@MXV@roo{0.0000001}\fi%
\FPupn\@MXV@transx{#1 \@MXV@roo{} \@MXV@viewx{} mul sub}%
\FPupn\@MXV@transy{#2 \@MXV@roo{} \@MXV@viewy{} mul sub}%
\FPupn\@MXV@transz{#3 \@MXV@roo{} \@MXV@viewz{} mul sub}%
```

Rotation matrix

```
\xdef\@MXV@matrix{%
 \@MXV@leftx\space\@MXV@lefty\space\@MXV@leftz\space%
 \@MXV@upx\space\@MXV@upy\space\@MXV@upz\space%
 \@MXV@viewx\space\@MXV@viewy\space\@MXV@viewz}%
```

Transformation matrix

```
\xdef\@MXV@matrix{%
 \@MXV@matrix\space\@MXV@transx\space%
 \@MXV@transy\space\@MXV@transz%
```

```
  }%
}% end of \@MXV@ciiwmatrix
```

Macro for parsing one line of 3D views file (old format)

```
\newcommand{\@MXV@parseline}[6][]{%
  \pdfstringdef\@MXV@xname{#1}% name of the view (optional)
  \ifthenelse{\equal{#2}{}}{%
    \xdef\@MXV@coo{0 0 0}%
  }{%
    \xdef\@MXV@coo{#2}%
  }%
  \ifthenelse{\equal{#3}{}}{%
    \xdef\@MXV@ctoc{0 -1 0}%
  }{%
    \xdef\@MXV@ctoc{#3}%
  }%
  \ifthenelse{\equal{#4}{}}{%
    \xdef\@MXV@roo{0}%
  }{%
    \xdef\@MXV@roo{#4}%
  }%
  \ifthenelse{\equal{#5}{}}{%
    \xdef\@MXV@roll{0}%
  }{%
    \xdef\@MXV@roll{#5}%
  }%
  \ifthenelse{\equal{#6}{}}{%
    \xdef\@MXV@aac{30}%
  }{%
    \xdef\@MXV@aac{#6}%
  }%
}
```

For parsing lines of views file (new format)

```
\define@key{MXV@view}{VIEW}[]{%
  \ifnum\@MXV@cursection<\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      A VIEW section cannot be nested into another section}{}%
  \fi%
  \def\@MXV@cursection{0}%
  \pdfstringdef\@MXV@xname{#1}% name of the view (optional)
  %default camera settings
  \gdef\@MXV@coo{0 0 0}%
  \gdef\@MXV@ctoc{0 -1 0}%
  \gdef\@MXV@roo{0}%
  \gdef\@MXV@roll{0}%
  \gdef\@MXV@aac{30}%
  %default background, lights, render mode
  \xdef\@MXV@background{/BG <</CS/DeviceRGB/C [\@MXV@defaultbg]>>}%
  \xdef\@MXV@lights{/LS <</Subtype/\@MXV@defaultlights>>}%
  \gdef\@MXV@render{/RM <</Subtype/\@MXV@defaultrender>>}%
  %initialise array of node dicts
  \gdef\@MXV@naarray{}%
  \global\@MXV@nodecount=\z@
  %initialise array of crosssection dicts
  \gdef\@MXV@saarray{}%
  \global\@MXV@cscount=\z@
}
\define@key{MXV@view}{COO}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      COO entry not allowed here; must go into a VIEW section}{}%
  \fi%
  \xdef\@MXV@coo{#1}%
}
\define@key{MXV@view}{C2C}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      C2C entry not allowed here; must go into a VIEW section}{}%
  \fi\xdef\@MXV@ctoc{#1}%
}
\define@key{MXV@view}{ROO}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      ROO entry not allowed here; must go into a VIEW section}{}%
  \fi\xdef\@MXV@roo{#1}%
}
\define@key{MXV@view}{AAC}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      AAC entry not allowed here; must go into a VIEW section}{}%
  \fi\xdef\@MXV@aac{#1}%
}
\define@key{MXV@view}{ROLL}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      ROLL entry not allowed here; must go into a VIEW section}{}%
  \fi\xdef\@MXV@roll{#1}%
}
\define@key{MXV@view}{BGCOLOR}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      BGCOLOR entry not allowed here; must go into a VIEW section}{}%
  \fi\def\@MXV@background{/BG<</CS/DeviceRGB/C[#1]>>}%
```

```
}
\define@key{MXV@view}{LIGHTS}{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      LIGHTS entry not allowed here; must go into a VIEW section}{}%
  \fi\def\@MXV@lights{/LS <</Subtype/#1>>}%
}
\define@key{MXV@view}{RENDERMODE}{%
  \ifnum\@MXV@cursection=\z@
    \def\@MXV@render{/RM <</Subtype/#1>>}%
  \else%
    \ifnum\@MXV@cursection=\@ne
      \def\@MXV@nrender{/RM <</Subtype/#1>>}%
    \else%
      \PackageError{rmannot}{%
        File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
          RENDERMODE entry not allowed here; must go into %
          either a VIEW or a PART section}{}%
    \fi%
  \fi%
}
\define@key{MXV@view}{PART}[]{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      PART not allowed here; must be a sub-section of %
      a VIEW section}{}%
  \fi%
  \ifthenelse{\equal{#1}{}}{%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      You must provide a valid PART name (PART=<part name>), as
      shown in the model tree of the 3D object %
      (go to ‘View’->‘Navigation Panels’->‘Model Tree’ %
      in Adobe Reader)}{}%
  }{}%
  \def\@MXV@cursection{1}%
  \pdfstringdef\@MXV@partname{#1}% name of the part
  \gdef\@MXV@nopacity{}%
  \gdef\@MXV@nvisibility{}%
  \gdef\@MXV@nrender{}%
  \gdef\@MXV@ntransform{}%
}
\define@key{MXV@view}{CROSSSECT}[]{%
  \ifnum\@MXV@cursection=\z@\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      CROSSSECT not allowed here; must be a sub-section of %
      a VIEW section}{}%
  \fi%
  \ifthenelse{\equal{#1}{}}{}{%
    \PackageWarning{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      CROSSECT does not take a value%
    }%
  }%
  \def\@MXV@cursection{2}%
  \gdef\@MXV@cscenter{0 0 0}%
  \gdef\@MXV@csorient{null 0 0}%
}
\define@key{MXV@view}{OPACITY}{%
  \ifnum\@MXV@cursection=\@ne\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      OPACITY entry not allowed here; must go into a PART section}{}%
  \fi%
  \gdef\@MXV@nopacity{/O #1}%
}
\define@key{MXV@view}{VISIBLE}{%
  \ifnum\@MXV@cursection=\@ne\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      VISIBLE entry not allowed here; must go into a PART section}{}%
  \fi%
  \gdef\@MXV@nvisibility{/V #1}%
}
\define@key{MXV@view}{MATRIX}{%
  \ifnum\@MXV@cursection=\@ne\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      MATRIX entry not allowed here; must go into a PART section}{}%
  \fi%
  \gdef\@MXV@ntransform{/M [#1]}%
}
\define@key{MXV@view}{CENTER}{%
  \ifnum\@MXV@cursection=2\relax\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: CENTER %
      entry not allowed here; must go into a CROSSECT section}{}%
  \fi%
  \gdef\@MXV@cscenter{#1}%
}
\define@key{MXV@view}{ORIENTATION}{%
  \ifnum\@MXV@cursection=2\relax\else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: ORIENT %
      entry not allowed here; must go into a CROSSECT section}{}%
  \fi%
  \gdef\@MXV@csorient{#1}%
```

```
}
\define@key{MXV@view}{END}[]{%
  \ifcase\@MXV@cursection%
    %END VIEW
    \edef\@MXV@args{%
      \@MXV@coo\space\@MXV@ctoc\space\@MXV@roo\space%
      \@MXV@roll\space}%
    \expandafter\@MXV@ciiwmatrix\@MXV@args% build C2W matrix
    \global\advance\@MXV@viewscount by \@ne%
    \ifthenelse{\equal{\@MXV@xname}{}}{% default view name
      \pdfstringdef\@MXV@xname{View \the\@MXV@viewscount}%
    }{}%
    \ifthenelse{\equal{\@MXV@naarray}{}}{}{%
      \gdef\@MXV@naentry{/NR true/NA [\@MXV@naarray]}%
    }%
    \ifthenelse{\equal{\@MXV@saarray}{}}{}{%
      \gdef\@MXV@saentry{/SA [\@MXV@saarray]}%
    }%
    \@MXV@viewobj% create pdf object of 3D view
    %append current view obj ref to VA array
    \xdef\@MXV@varray{\@MXV@varray\space\@MXV@@viewobj}%
    \global\@MXV@viewsprovidedtrue%
    \def\@MXV@cursection{-1}%
  \or%
    %END PART
    \global\advance\@MXV@nodecount by \@ne
    \@MXV@nodeobj% create pdf object of 3D node dict
    %append it to node array
    \xdef\@MXV@naarray{\@MXV@naarray\space\@MXV@@nodeobj}%
    \def\@MXV@cursection{0}%
  \or%
    %END CROSSSECT
    \global\advance\@MXV@cscount by \@ne
    \@MXV@csobj% create pdf object of 3D cross section dict
    %append it to cross section array
    \xdef\@MXV@saarray{\@MXV@saarray\space\@MXV@@csobj}%
    \def\@MXV@cursection{0}%
  \else%
    \PackageError{rmannot}{%
      File \@MXV@viewsfileii, line \the\@MXV@inputlineno: %
      There is nothing to be ENDed here}{}%
  \fi%
}%
```

Macro for generating an array of 3D views (varray)

```
\def\@MXV@procinputline#1{\setkeys{MXV@view}{#1}}
\newcount\@MXV@inputlineno
\def\@MXV@buildva{%
    \global\@MXV@viewscount=0\relax% dps
    \xdef\@MXV@varray{}% empty varray
```

```
%
%default view (one of the command options 3Dcoo, 3Dc2c, etc. given)
  \edef\@MXV@args{%
    \@MXV@coo\space\@MXV@ctoc\space\@MXV@roo\space%
    \@MXV@roll\space}%
  \expandafter\@MXV@ciiwmatrix\@MXV@args% build C2W matrix
  \pdfstringdef\@MXV@xname{Default}%
  \if@MXV@defaultviewprovided%
    \@MXV@viewobj% create pdf object of 3D view
%      \edef\@MXV@defaultview{/3DV \@MXV@@viewobj}%
    \edef\@MXV@defaultview{\@MXV@@viewobj}%
  \fi%
```

Read out 3D views file (new version)

```
  \def\@MXV@cursection{-1}% views file is divided in sections
  \IfFileExists{\@MXV@viewsfileii}{%
    \begingroup%
    \endlinechar=-1% suppress trailing space at input line end
    \@MXV@inputlineno=\z@%
    \openin\@MXV@@viewsfile=\@MXV@viewsfileii%
    \read\@MXV@@viewsfile to \@MXV@inputline%
    \ifeof\@MXV@@viewsfile\setboolean{@MXV@eof}{true}\else%
      \setboolean{@MXV@eof}{false}\fi%
    \whiledo{\not\boolean{@MXV@eof}}{%
      \advance\@MXV@inputlineno by \@ne%
      %process input line
      \edef\@MXV@@inputline{{\@MXV@inputline}}%
      \expandafter\@MXV@procinputline\@MXV@@inputline%
      \read\@MXV@@viewsfile to \@MXV@inputline%
      \ifeof\@MXV@@viewsfile%
        \setboolean{@MXV@eof}{true}%
      \else%
        \setboolean{@MXV@eof}{false}%
      \fi%
    }%
    \closein\@MXV@@viewsfile%
    \endgroup%
  }{}%
```

Make the first view in the VA array the default view, if no default one has explicitly been provided, but if the VA array itself is empty too (no additional views provided) use our fallback view (c2c=0 -1 0) as default

```
  \ifthenelse{\not\boolean{@MXV@defaultviewprovided}%
    \and\boolean{@MXV@viewsprovided}}{%
%      \xdef\@MXV@defaultview{/3DV/F}%
    \xdef\@MXV@defaultview{/F}%
  }{}%
  \ifthenelse{\not\boolean{@MXV@defaultviewprovided}%
    \and\not\boolean{@MXV@viewsprovided}}{%
    \@MXV@viewobj% create pdf object of 3D view
%      \edef\@MXV@defaultview{/3DV \@MXV@@viewobj}%
   \edef\@MXV@defaultview{\@MXV@@viewobj}%
  }{}%
}
```

Following macros, including the 3D inclusion macro have driver specific implementations dvips versions macro for creating 3D view object and associated projection dict

```
\def\@MXV@viewobj{\literalps@out{%
%projection dict
  \ps@mark/_objdef {pdict\therm@Cnt_\the\@MXV@viewscount}%
    /type/dict/OBJ pdfmark^^J%
  \ps@mark{pdict\therm@Cnt_\the\@MXV@viewscount} <<%
    /Subtype/P/FOV \@MXV@aac/PS/Min>>/PUT pdfmark^^J%
  \ps@mark/_objdef {viewobj\therm@Cnt_\the\@MXV@viewscount}%
    /type/dict/OBJ pdfmark^^J%
  \ps@mark{viewobj\therm@Cnt_\the\@MXV@viewscount} <<%
    /MS/M%
    /CO \@MXV@roo%
    /P {pdict\therm@Cnt_\the\@MXV@viewscount}%
    /C2W[\@MXV@matrix]%
    /XN(\@MXV@xname)%
    /IN(\@MXV@xname)%
        \@MXV@background%
        \@MXV@lights%
        \@MXV@render%
        \@MXV@naentry%
        \@MXV@saentry%
    >>%
  /PUT pdfmark
}%
\xdef\@MXV@@viewobj{{viewobj\therm@Cnt_\the\@MXV@viewscount}}%
}%
%3D node object
\def\@MXV@nodeobj{\literalps@out{%
  \ps@mark/type/dict%
    /_objdef {nodeobj\therm@Cnt_\the\@MXV@viewscount_%
       \the\@MXV@nodecount}/OBJ pdfmark^^J%
  \ps@mark{nodeobj\therm@Cnt_\the\@MXV@viewscount_%
     \the\@MXV@nodecount}<<%
      /Type/3DNode%
      /N (\@MXV@partname)%
      \@MXV@nopacity\@MXV@nvisibility\@MXV@ntransform%
        \@MXV@nrender%
    >>/PUT pdfmark
}%
\xdef\@MXV@@nodeobj{%
  {nodeobj\therm@Cnt_\the\@MXV@viewscount_%
    \the\@MXV@nodecount}}%
}%
```

```
%3D cross section object
\def\@MXV@csobj{\literalps@out{%
  \ps@mark/type/dict%
      /_objdef {csobj\therm@Cnt_\the\@MXV@viewscount_%
      \the\@MXV@cscount}/OBJ pdfmark^^J%
  \ps@mark{csobj\therm@Cnt_\the\@MXV@viewscount_%
      \the\@MXV@cscount}<<%
          /Type/3DCrossSection%
          /C [\@MXV@cscenter]%
          /O [\@MXV@csorient]%
      >>/PUT pdfmark
  }%
  \xdef\@MXV@@csobj{%
    {csobj\therm@Cnt_\the\@MXV@viewscount_\the\@MXV@cscount}}%
}%
```

MXV@user family

```
\define@key{MXV@user}{3Dbg}[1 1 1]{%
  \def\@MXV@defaultbg{#1}%
  \def\@MXV@background{/BG<</CS/DeviceRGB/C[#1]>>}%
}
\define@key{MXV@user}{3Djscript}{%
  \def\rma@rmAnnot@iiiDjs{#1}%
  \ifx\rma@rmAnnot@iiiDjs\@empty\let\rma@addResources\@empty
    \let\rma@addFileSpecs\@empty\else
```

We process resources when there are some to process :-)

```
    \rma@toks={}\def\rmiiid@addToScriptsArray{}%
    \@for\rma@arg:=\rma@rmAnnot@iiiDjs\do{%
      \rma@edefexecute{\noexpand
        \filename@parse{\rma@useNamedPath{\rma@arg}}}%
      \@ifundefined{filename@ext}{%
        \rma@PkEr@iii{\rma@useNamedPath{\rma@arg}}}{}%
      \edef\rmiiid@addToScriptsArray{\rmiiid@addToScriptsArray
        \ps@mark{jscriptiiid\therm@Cnt}%
        {rmfilespec\therm@Cnt-JS\rma@arg}%
        /APPEND pdfmark^^J%
      }%
      \edef\rma@fs@expand{rmFileStrm\rma@arg}%
      \@ifundefined{\rma@fs@expand}{%
      \rm@csarg\xdef{\rma@fs@expand}%
        {rmfstream\therm@Cnt-JS\rma@arg}%
        \def\rma@embed{1}}{\def\rma@embed{0}}%
      \edef\rma@tmp@exp{\the\rma@toks%
        \noexpand\\{JS\rma@arg}%
        {\filename@area}{\filename@base.\filename@ext}%
        {\rma@embed}{\csname\rma@fs@expand\endcsname}%
        {\rm@csarg\noexpand{rma@mt@\rma@arg}}}%
      \rma@toks=\expandafter{\rma@tmp@exp}%
    }% do
    \let\\\rm@appendNameTree
    \expandafter\xdef\expandafter\rma@addResources%
      \expandafter{\the\rma@toks}%
    \let\\\rm@appendFileSpecs
    \expandafter\xdef\expandafter\rma@addFileSpecs%
      \expandafter{\the\rma@toks}%
  \fi
}
\define@key{MXV@user}{3Dcoo}{%
  \def\@MXV@coo{#1}%
  \setboolean{@MXV@defaultviewprovided}{true}%
}
\define@key{MXV@user}{3Dc2c}{%
  \def\@MXV@ctoc{#1}%
  \setboolean{@MXV@defaultviewprovided}{true}%
}
\define@key{MXV@user}{3Droo}{%
  \def\@MXV@roo{#1}%
  \setboolean{@MXV@defaultviewprovided}{true}%
}
\define@key{MXV@user}{3Daac}{%
  \def\@MXV@aac{#1}%
  \setboolean{@MXV@defaultviewprovided}{true}%
}
\define@key{MXV@user}{3Droll}{%
  \def\@MXV@roll{#1}%
  \setboolean{@MXV@defaultviewprovided}{true}%
}
```

Since we are starting fresh, we don't use the old format used by movie15, so I am renaming 3Dviews2 to 3Dviews and eliminating the old format and code completely.

```
\define@key{MXV@user}{3Dviews}{%
  \IfFileExists{#1}{%
    \def\@MXV@viewsfileii{#1}%
  }{%
    \PackageError{rmannot}{3D views  file `#1' cannot be opened%
    }{%
      Make sure file `#1' exists and is readable!%
    }%
  }%
}
\define@choicekey+{MXV@user}{3Dlights}%
  {None,White,Day,Night,Hard,Primary,Blue,%
    Red,Cube,CAD,Headlamp}[Cube]{%
  \gdef\@MXV@defaultlights{#1}%
  \gdef\@MXV@lights{/LS <</Subtype/#1>>}%
}{\PackageWarning{rmannot}{Bad choice for 3Dlights, permissible
  values are None, White, Day, Night, Hard, Primary, Blue,
  Red, Cube, CAD, HeadLamp. Try again}}
```

```
\define@choicekey+{MXV@user}{3Drender}%
  {Solid,SolidWireframe,Transparent,TransparentWireframe,%
   BoundingBox,TransparentBoundingBox,TransparentBoundingBoxOutline,%
   Wireframe,ShadedWireframe,HiddenWireframe,Vertices,ShadedVertices,%
   SolidOutline,Illustration,ShadedIllustration}[Solid]{%
  \gdef\@MXV@defaultrender{#1}%
  \gdef\@MXV@render{/RM <</Subtype/#1>>}%
}{\PackageWarning{rmannot}{Bad choice for 3Dlights, permissible
  values are Solid, SolidWireframe, Transparent, TransparentWireframe,
  BoundingBox, TransparentBoundingBox, TransparentBoundingBoxOutline,
  Wireframe, ShadedWireframe, HiddenWireframe, Vertices,
  ShadedVertices, SolidOutline, Illustration, ShadedIllustration.
  Try again}}
⟨/3Dcode⟩
⟨*package⟩
\rma@input@iiidCode
⟨/package⟩
```

5 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

6 Change History

v1.0 (2010/09/24)
General: Made a correction to a typo for CuePoints (I had Cuepoints). Now the cue points feature works. Removed some spurious spaces from \rmAnnot 2

v1.0a (2010/09/29)
General: Added an invisible option for \rmAnnot. The annot is given a transparent poster, if no poster is specified. 2

v1.0b (2010/09/29)
General: Experimental use of VideoPlayerPlus.swf and VideoPlayerX.swf . 2

v1.0c (2010/10/01)
General: Added \let\rma@addResources\@empty \let\rma@addFileSpecs\@emptyto clean out these macros, if one \rmAnnot had resources, these were included in the next \rmAnnot as well causing distiller to cough a fur ball. The problem manifests itself when \useVideoPlayerPlus or \useVideoPlayerX is used. 2

v1.1 (2010/10/07)
General: Added definitions for skins so they can be referenced by name in the resources key. Added support for \Name and \urlName inside a eform field. 2

v1.1a (2010/17/07)
General: Added definitions for none and noChange that can be recognized by JS to indicate that the video should have no skin, and the video should have the same skin as the previous video (noChange) 2

v1.1b (2010/14/12)
General: Corrected the definition of borderwidth. Acrobat uses for none, thin, medium, and thick the values of 0,1,3,5 not 0,1,2,3 (for some reason). 2

v1.1d (2010/27/12)
General: Made some minor changes, labeled as 12/27/10 (2010/22/12 v1.1c) Made the definition \saveNamedPath global, needed for the fldigigal package. 2

v2.0.6 (2018/03/21)
\pathToPlayers: Placed \pathToSkins in a group and disallowed unicode encoding by hyperref 4

v2.0a (2011/09/11)
\pathToPlayers: Arguments now pass through

\pdfstringdef 4

v2.0b (2015/09/30)

\pathToPlayers: Added \AcroVer and a more intelligent method of finding the path to skins and players. 5

v2.0d (2016/10/09)

General: Added 10.4 function \mmShowLoopButton 8

\useVideoPlayerPlus: Removed support for VideoPlayerPlus.swf. 7

v2.2 (2020/08/21)

General: Add user interface to design of default poster 39

Added all skin name 17

Added defaultposter designed for use with MP3s 16

Added an all skin 11

\pathToPlayers: Added 32 and 64 as values of the win key of \AcroVer 5

\rmAnnot: Added pmpv to \rmAnnot, now displays name of the file to play 24

Support for keys width and height of the rmAnnot family. The scale key is also defined. 25