The digicap-pro Package

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

processed May 23, 2018

Contents

1 ⟨*package⟩

1 Introduction

This is a package that can create fancy, transparent captions to photos,[1] or any graphics file. The captions can be set to be continually visible, or only visible on roll-over. Captions can be placed horizontally and vertically with optional arguments. A visible, possibly transparent, border can be placed around the caption as well.

2 Package Options

`display1` The `display1` option is used to create a photo album, a single PDF that contains many photos with captions which are accessible by clicking thumbnails.

2 `\DeclareOption{display1}{\AtEndOfPackage{\dc@input@displayi}}`

[1] Transparent here means having an opacity between 0 and 1.

```
\def\dc@input@displayi{\InputIfFileExists{digi-p1.def}{}{}}
\ProcessOptions
```

3 Code

Let's get this show on the road!

3.1 Required Packages

The package builds on packages developed as part of AeB or AeB Pro:

- aeb_pro: supplies support for layers and JavaScript management of layers
- graphicxbox: places a graphic as the background of a box
- opacity-pro: creates the transparency effects
- eforms: use to create Acrobat form buttons with a roll-over action to make roll-over captions visible or hidden.

The graphicx package is also used to import digital photos, or other graphics.

```
\RequirePackage{eforms}
\RequirePackage{graphicx}
\RequirePackage{graphicxbox}
\RequirePackage{opacity-pro}
```

3.2 \opcolorbox

The following are the definitions of the key-value pairs used by `\opcolorbox`. A brief description of their purpose appears in the section devoted to `\opcolorbox`.

```
\def\dc@nocolor{nocolor}
\define@key{opcolorbox}{borderwidth}[2pt]{\def\opcb@borderwidth{#1}}
\define@key{opcolorbox}{fboxsep}[6pt]{\def\opcb@fboxsep{#1}}
\define@key{opcolorbox}{width}[\linewidth]{\def\opcb@width{#1}}
\define@key{opcolorbox}{bordercolor}[black]{\def\opcb@bordercolor{#1}}
\define@key{opcolorbox}{bgcolor}[white]{\def\opcb@bgcolor{#1}}
\define@key{opcolorbox}{borderop}[.5]{\def\opcb@borderop{#1}}
\define@key{opcolorbox}{bordertextop}[1]{\def\opcb@bordertextop{#1}}
\define@key{opcolorbox}{bgop}[.5]{\def\opcb@bgop{#1}}
\define@key{opcolorbox}{textop}[1]{\def\opcb@textop{#1}}
\define@key{opcolorbox}{borderblend}[Normal]{%
    \def\opcb@borderblendmode{#1}}
\define@key{opcolorbox}{bgblend}[Normal]{\def\opcb@bgblendmode{#1}}
\setkeys{opcolorbox}{borderwidth,fboxsep,width,bordercolor,bgcolor,%
    bordertextop,borderop,bgop,textop,borderblend,bgblend}
```

`\opcolorbox` A general purpose color box that creates two color boxes, a larger one with a smaller one centered vertically and horizontally inside the larger one. Transparent options allow separate control over the opacity settings of the larger and smaller rectangle as well as the text that is written within the smaller rectangle.

Optional key-values for the first parameter

`borderwidth`: The border width. The default is `2pt`

`fboxsep`: The space between the border and the text, the default is `6pt`

`width`: The width of `\parbox`, the default is `\linewidth`

`bordercolor`: A named color of border, the default is `black`. A special value of `nocolor` is recognized, in that case, no color is applied.

`bgcolor`: A named color of background, the default is `white`. A special value of `nocolor` is recognized, in that case, no color is applied.

`borderop`: A number type, the opacity for border $0 \leq$ number ≤ 1, the default is .5

`bgop`: A number type, the opacity for background $0 \leq$ number ≤ 1, the default is .5

`textop`: A number type, the opacity for text $0 \leq$ number ≤ 1, the default is 1

`borderblend`: The blend mode for the border, the default is `Normal`

`bgblend`: The blend mode for the background, the default is `Normal`

Second parameter, required. The text that goes within the box.

```
\def\dc@mark{[\space}%]
\newcommand{\opcolorbox}[2][]{\begingroup
  \edef\dc@tmp@exp{\noexpand\setkeys{opcolorbox}{#1}}\dc@tmp@exp
  \ifx\opcb@bgcolor\dc@nocolor\let\opcb@set@bgcolor\mbox
  \else\def\opcb@set@bgcolor{\colorbox{\opcb@bgcolor}}\fi
  \ifx\opcb@bordercolor\dc@nocolor\let\opcb@set@bordercolor\mbox
  \else\def\opcb@set@bordercolor{\colorbox{\opcb@bordercolor}}\fi
  \setlength{\fboxsep}{\opcb@borderwidth}\setlength{\fboxrule}{0pt}%
  \begin{settransparency}[\opcb@borderblendmode]{\opcb@bordertextop}%
    {\opcb@borderop}%
    \opcb@set@bordercolor{\parbox[c]{\opcb@width}{%
    \setlength{\fboxsep}{\opcb@fboxsep}\setlength{\fboxrule}{0pt}%
    \begin{settransparency}[\opcb@bgblendmode]{\opcb@textop}%
      {\opcb@bgop}%
      \opcb@set@bgcolor{\parbox[c]{\linewidth-2\fboxsep}{%
```

for dvips, CA is not recognized since dvips does not stroke backgrounds, so we insert another layer of transparency, with ca=CA=`\opcb@textop`

```
      \begin{settransparency}{\opcb@textop}{\opcb@textop}%
        #2%
      \end{settransparency}%
      }}%
    \end{settransparency}}}%
  \end{settransparency}%
    \endgroup
}
```

3.3 \digiCap and \digiCap*

The \digiCap command is defined in this section; there is an * option that changes the caption into a rollover. Before we get started, we define several commands that support some of the options for this command.

The \dc@vCaptionPlacement command accomplishes two things: It records the document author's choice for vertical placement of the caption (saving it in \aeb@captionPlacement), and calculates the amount of vertical displacement needed to overlay the rollover form field correctly over the picture. Possible values for #1 and b, c, and t. The default is b. This command is called when the author sets vcaption, a key belonging to the dc@commands family. Defined below.

```
\def\dc@vCaptionPlacement#1{\def\dc@captionPlacement{#1}%
    \def\dc@@captionPlacement{0pt}%
    \if\dc@captionPlacement c%
        \def\dc@@captionPlacement{-\dc@graphicHalfHeight+3pt}\else
    \if\dc@captionPlacement t%
        \def\dc@@captionPlacement{-\dc@graphicHeight}%
    \else
        \def\dc@captionPlacement{b}%
        \def\dc@@captionPlacement{0pt}%
    \fi\fi
}
\dc@vCaptionPlacement{b}
```

We set the horizontal placement of the caption, possible values are l, c, and r. The default is c. This command is called when the author sets hcaption, a key belonging to the dc@commands family. Defined below.

```
\def\dc@hCaptionPlacement#1{\def\dc@argi{#1}%
    \if\dc@argi l\def\dc@Hplacement{\relax}\else
        \if\dc@argi c\def\dc@Hplacement{\hfil}\else
            \if\dc@argi r\def\dc@Hplacement{\hfill}\else
                \def\dc@Hplacement{\relax}\fi\fi\fi
}
\dc@hCaptionPlacement{c}
```

```
\def\dc@calc@adj@width#1{{%
    \edef\dc@tmp@exp{\noexpand\setkeys{opcolorbox}{#1}}\dc@tmp@exp
    \setlength{\linewidth}{\dc@graphicWidth-2\fboxsep}%
    \setlength{\dimen@}{\opcb@width}%
    \setlength{\dimen@ii}{\opcb@borderwidth}%
    \setlength{\dimen@}{\dimen@-2\dimen@ii}%
    \xdef\dc@adj@width{\the\dimen@}%
}}
```

This is the command that inserts the caption

```
#1=path to graphic
#2=box content (#3-#5 are included in the box content)
#3=KVPairs of \opcolorbox
#4=content of \opcolorbox
#5=either empty or \eBld, if layers used
```

```
\long\def\dc@insert@graphicx@opcolor@boxes#1#2#3#4#5{%
    \graphicxbox{#1}{#2\parbox[\dc@captionPlacement]%
        [\dc@graphicHeight-2\fboxsep]{\dc@graphicWidth-2\fboxsep}%
        {\vskip0pt\dc@Hplacement\opcolorbox[#3,width=\dc@adj@width]%
        {#4}\par\kern0pt}#5}%
}
```

\graphicHeight
\graphicWidth

These two commands may be used within the `<caption>` argument of the command `\digiCap`. `\graphicHeight` is used to set the height of a `minipage` of `\parbox` for a vertically oriented caption. An example appears in the demo file.

```
\def\graphicHeight{\dc@graphicHeight-2\fboxsep-%
    \dc@outerboxsep-\dc@outerboxsep-\opcb@borderwidth-\opcb@borderwidth}
\def\graphicWidth{\dc@graphicWidth-2\fboxsep-%
    \dc@outerboxsep-\dc@outerboxsep-\opcb@borderwidth-\opcb@borderwidth}
```

\digiCap

A command that places a picture as background of a box, and places a, possibly, transparent caption, optionally, with a border. The syntax is...

```
\digiCap[<dc@commands_kvps>]
      {<file>}[<opcolorbox_fam_kvps>]{<caption>}
```

or

```
\digiCap*[<dc@commands_kvps>]
      {<file>}[<opcolorbox_fam_kvps>]{<caption>}
```

If the optional `*` appears, then `\dc@digiCapRollover` is called, otherwise, `\dc@digiCap`.

The `dc@commands xkeyval` family is defined below.

Optional key-values for the first parameter. This set of parameters control the placement of the caption on top of the background picture. There is also a parameter to set the `\includegraphics` options, and the underlying form field name, in the case of a rollover.

`outerboxsep:` The space the surrounds the boundary of the caption, the default is `3pt`

`vcaption:` The vertical placement of the caption on the background graphic, possible values are `b`, `c`, and `t`. The default is `b`.

`hcaption:` The horizontal placement of the caption on the background graphic, possible values are `l`, `c`, and `r`. The default is `c`.

`inclgraphicx:` The value of this key is a list of key-value pairs that are passed on to the underlying `\includegraphics` command.

`rollovername:` The basename of the push button form field that is used for a rollover effect. This command is used only with `\digiCap*`, ignored otherwise. For the `\digiCap*` command, this key is optional, if not present, this package supplies a name.

Second parameter, required. The second parameter `<file>` is the path to the graphic to be use as a background to this box.

Optional key-values for the third parameter. The options for the underlying `\opcolorbox`. See above for a listing and description.

Fourth parameter, required. The content of the caption.

```
\define@key{dc@commands}{outerboxsep}[3pt]{\def\dc@outerboxsep{#1}}
\define@key{dc@commands}{vcaption}[b]{\dc@vCaptionPlacement{#1}}
\define@key{dc@commands}{hcaption}[c]{\dc@hCaptionPlacement{#1}}
\define@key{dc@commands}{inclgraphicx}[]{%
    \def\dc@inclgraphicx{#1\dc@incgfx@addkeys}}
\let\dc@incgfx@addkeys\@empty
\define@key{dc@commands}{rollovername}[]{{%
    \gdef\dc@rollovername{#1}%
    \ifx\dc@rollovername\@empty
        {\count0=\dc@rollover@cnt\advance\count0by1\relax
        \xdef\dc@rollover@cnt{\the\count0}%
        \xdef\dc@rollovername{Cnt\dc@rollover@cnt}}%
    \fi
}}
\let\dc@rollovername\@empty
\def\dc@rollover@cnt{0}%
\setkeys{dc@commands}{outerboxsep,vcaption,hcaption,inclgraphicx}
\newcommand{\digiCap}{\@ifstar{\dc@digiCapRollover}
    {\dc@digiCap}}
```

`\dc@digiCap` This creates a digital photo with caption, no rollover.

```
\newcommand{\dc@digiCap}[2][]{\begingroup
    \edef\dc@tmp@exp{\noexpand\setkeys{dc@commands}{#1}}\dc@tmp@exp
    \def\dc@filename{#2}\setlength{\fboxsep}{\dc@outerboxsep}%
    \dc@@digiCap
}
\newcommand{\dc@@digiCap}[2][]{%
    \edef\dc@tmp@exp{\noexpand\setkeys{Gin}{\dc@inclgraphicx}}%
    \dc@tmp@exp\edef\dc@tmp@exp{\setbox0=
        \hbox{\noexpand\includegraphics[draft,\dc@inclgraphicx]%
        {\dc@filename}}}\dc@tmp@exp\dimen@=\dp0 \advance\dimen@\ht0
    \edef\dc@graphicHeight{\the\dimen@}%
    \edef\dc@graphicWidth{\the\wd0}%
    \dc@calc@adj@width{#1}\parbox{\dc@graphicWidth}{%
    \dc@insert@graphicx@opcolor@boxes{\dc@filename}{}{#1}{#2}{}}%
\endgroup}
```

`\dc@digiCap*` Same as `\dc@digiCap`, but the caption is placed in a layer and a rollover effect is use to make the caption appear. The syntax is...

```
\newcommand{\dc@digiCapRollover}[2][]{\begingroup
```

```
    \edef\dc@tmp@exp{\noexpand\setkeys{dc@commands}{#1}}%
    \dc@tmp@exp\def\dc@filename{#2}%
    \setlength{\fboxsep}{\dc@outerboxsep}%
    \ifx\dc@rollovername\@empty\setkeys{dc@commands}{rollovername}\fi
    \dc@@digiCapRollover
}
\newcommand{\dc@@digiCapRollover}[2][]{%
    \edef\dc@tmp@exp{\noexpand\setkeys{Gin}{\dc@inclgraphicx}}%
    \dc@tmp@exp\edef\dc@tmp@exp{\setbox0=\hbox{%
        \noexpand\includegraphics[draft,\dc@inclgraphicx]%
        {\dc@filename}}}\dc@tmp@exp\dimen@=\dp0\advance\dimen@\ht0
    \edef\dc@graphicHeight{\the\dimen@}\dimen@=.5\dimen@
    \edef\dc@graphicHalfHeight{\the\dimen@}%
    \edef\dc@graphicWidth{\the\wd0}%
    \dc@calc@adj@width{#1}\parbox{\dc@graphicWidth}{%
        \raisebox{\dc@@captionPlacement}[0pt][0pt]{\rlap
        {\pushButton[\presets{\digiCapsPresets{\dc@rollovername}}%
         \presets{\hiddenPresets}]{dcRollover.\dc@rollovername}%
         {\dc@graphicWidth}{\dc@graphicHeight}}}%
         \dc@insert@graphicx@opcolor@boxes{\dc@filename}%
        {\xBld{\dc@rollovername}}{#1}{#2}{\eBld}%
    }\global\let\dc@rollovername\@empty%
\endgroup}
```

This is a listing of options to be used by the push button that supplies the rollover effect.

```
\def\digiCapsPresets#1{\W0\BG{}\BC{}\H{N}\autoCenter{n}% rollover mod
    \AA{\AAMouseEnter{\JS{toggleSetThisLayer("#1",true);}}%
    \AAMouseExit{\JS{toggleSetThisLayer("#1",false);}}}}
\def\hiddenPresets{}
```

⟨/package⟩

⟨∗digidisplay1⟩

3.4 A Layout for Digital Display (Photo album)

This segment of code provides for a layout to display digital images. Thumbnails of the images are lined up in rows or columns. When the user rolls over a thumb, a large version of that photo appears in the display area. The photos can optionally contain a short caption, and a longer caption. This latter caption appears on a transparent background on top of the photo (`\digiCap` is used here).

3.4.1 Embedding Images/Creating Thumb Appearances

Embed each image using `\embedEPS`, then create other images of that digital in normal, rollover and down appearances. These are used for the form field thumbnails.

\PicsThisDoc This command is executed in the preamble of the document. The one argument is a comma delimited list of four parameters:

```
\PicsThisDoc
{%
    {<embed_name>}{<graphic_path>}{<short_caption>}{<long_caption>},
    {<embed_name>}{<graphic_path>}{<short_caption>}{<long_caption>},
    ...
    {<embed_name>}{<graphic_path>}{<short_caption>}{<long_caption>}
}
```

We pass each set of four arguments on to \dc@setPicsAndCaptions for processing.

```
\newcommand{\PicsThisDoc}[1]{%
    \@for\@args:=#1\do{\expandafter\dc@setPicsAndCaptions\@args}%
}
```

This command takes the four arguments passed to it from \PicsThisDoc, and passes the required args to the commands \dc@embedEPSCreateAppearances and to \dc@defTheseCaptions.

```
\def\dc@setPicsAndCaptions#1#2#3#4{%
    \dc@embedEPSCreateAppearances{#1}{#2}%
    \dc@defTheseCaptions{#1}{#3}{#4}%
}
```

This command embeds the graphic file #2, names that file as #1. The name #1 is later used to show the figure. The grahicxsp package is used here. This command also builds images used in the appearance states of the thumbnail images. The appearance states can be redefined, as desired.

\setThumbAppearances Set the appearances of the thumbnail images. There are three appearances: normal, rollover and push. The settings for these parameters are use in the command \dc@embedEPSCreateAppearances.

There is one optional argument, the value of this optional argument is the name of one of the photos; in this case, the second argument is used only for that picture. This way, you can change the appearance of the thumbs. Normally, they would all look the same.

```
\newcommand{\setThumbAppearances}[2][]{%
    \def\dc@argi{#1}\ifx\dc@argi\@empty\def\dc@thumbApprs{#2}%
    \setkeys{dc@ro@appr}{#2}\else
    \expandafter\def\csname dc@thumbApprs@#1\endcsname{#2}\fi
}
\define@key{dc@ro@appr}{normalop}[.5]{\def\dc@ro@appr@normalopacity{#1}}
\define@key{dc@ro@appr}{rolloverop}[1]%
    {\def\dc@ro@appr@rolloveropacity{#1}}
\define@key{dc@ro@appr}{downop}[.3]%
    {\def\dc@ro@appr@downopacity{#1}}
\define@key{dc@ro@appr}{boundarywidth}[30]%
    {\def\dc@ro@appr@boundarywidth{#1}}
\define@key{dc@ro@appr}{rgbcolor}[]{%
  \def\dc@ro@appr@rgbcolor{#1}\ifx\dc@ro@appr@rgbcolor\@empty
```

```
  \else
    \expandafter\ef@isitnamed\dc@ro@appr@rgbcolor\ef@nil
    \ifx\ef@latex@color\ef@y\expandafter
      \HyColor@XZeroOneThreeFour
        \expandafter{\dc@ro@appr@rgbcolor}{\dc@ro@appr@rgbcolor}{}{}%
      \edef\dc@ro@appr@rgbcolor{\dc@ro@appr@rgbcolor}\fi
\fi}
\let\dc@ro@appr@rgbcolor\@empty
%    {\def\dc@ro@appr@rgbcolor{#1}}
\define@key{dc@ro@appr}{cmykcolor}[0 0 1 0]{%
  \def\dc@ro@appr@cmykcolor{#1}\ifx\dc@ro@appr@cmykcolor\@empty
  \else
    \expandafter\ef@isitnamed\dc@ro@appr@cmykcolor\ef@nil
    \ifx\ef@latex@color\ef@y\expandafter
      \HyColor@XZeroOneThreeFour
        \expandafter{\dc@ro@appr@cmykcolor}{\dc@ro@appr@cmykcolor}{}{}%
      \edef\dc@ro@appr@cmykcolor{\dc@ro@appr@cmykcolor}\fi
\fi}
\def\dc@ro@appr@cmykcolor{0 0 1 0}
%    {\def\dc@ro@appr@cmykcolor{#1}}
```

Set the default values for these key-value pairs.

```
\setThumbAppearances{normalop,rolloverop,downop,boundarywidth,%
    rgbcolor,cmykcolor}
```

This command embeds the graphic file `#2`, names that file as `#1`. The name `#1` is later used to show the figure. The grahicxsp package is used here. This command also builds images used in the appearance states of the thumbnail images. The appearance states can be redefined, as desired.

```
\def\dc@embedEPSCreateAppearances#1#2{%
    \embedEPS[transparencyGroup]{#1}{#2}%
    \@ifundefined{dc@thumbApprs@#1}{\edef\dc@tmp@exp%
        {\noexpand\setkeys{dc@ro@appr}{\dc@thumbApprs}}}%
        {\edef\dc@tmp@exp{\noexpand\setkeys{dc@ro@appr}%
        {\csname dc@thumbApprs@#1\endcsname}}}\dc@tmp@exp
    \begin{createImage}{\bboxOf{#1}}{n#1}
        gsave
        \dc@mark/ca \dc@ro@appr@normalopacity
          /SetTransparency pdfmark
        \urxOf{#1} .1 mul \uryOf{#1} .1 mul moveto
        currentpoint translate
       .8  .8  scale
        \dc@mark{#1} /SP pdfmark
        grestore
    \end{createImage}
    \begin{createImage}{\bboxOf{#1}}{r#1}
        \dc@mark/ca \dc@ro@appr@rolloveropacity
          /CA \dc@ro@appr@rolloveropacity
          /SetTransparency pdfmark
        \dc@mark{#1} /SP pdfmark
```

```
        \dc@ro@appr@boundarywidth\space setlinewidth
        \ifx\dc@ro@appr@rgbcolor\@empty
        \dc@ro@appr@cmykcolor\space setcmykcolor\else
        \dc@ro@appr@rgbcolor\space setrgbcolor\fi\space
        currentlinewidth 2 div dup
        \urxOf{#1} currentlinewidth sub \uryOf{#1} currentlinewidth sub
        rectstroke
    \end{createImage}
    \begin{createImage}{\bboxOf{#1}}{d#1}
        \dc@mark/ca \dc@ro@appr@downopacity
          /CA \dc@ro@appr@downopacity/SetTransparency pdfmark
        \dc@mark{#1} /SP pdfmark
        \dc@ro@appr@boundarywidth\space setlinewidth
        \ifx\dc@ro@appr@rgbcolor\@empty
        \dc@ro@appr@cmykcolor\space setcmykcolor\else
        \dc@ro@appr@rgbcolor\space setrgbcolor\fi\space
        currentlinewidth 2 div dup
        \urxOf{#1} currentlinewidth sub \uryOf{#1} currentlinewidth sub
        rectstroke
    \end{createImage}
}
```

This command takes that short and long captions and saves them in a text macro under the name `#1Caption` and `#1Text`, where `#1` is the graphic name.

```
\def\dc@defTheseCaptions#1#2#3{%
    \expandafter\gdef\csname #1Caption\endcsname{#2}%
    \expandafter\gdef\csname #1Text\endcsname{#3}%
}
```

3.4.2 Placing the elements on the page

This section of the code is devoted to defining the commands to insert the various elements on the page: the photos, the captions, and the thumbs.

`\presentationOrder` A command to create a text macro. The argument is a comma delimited list of photo names.

```
\newcommand{\presentationOrder}[1]{\def\dc@presentationOrder{#1}}
```

`\dcFirstOpt` `\dcSecondOpt` These two commands are used to pass optional arguments to `\digiCap`. \ with various required arguments). These controls for the appearance, transparancy, and positioning of the long caption box. These `\dcFirstOpt` and `\dcSecondOpt` are passed as the first optional parameter and third parameters of the `\digiCap` command.

```
\newcommand{\dcFirstOpt}[2][]{%
    \def\dc@argi{#1}\ifx\dc@argi\@empty\def\dc@icontrol{#2}\else
    \expandafter\def\csname dc@icontrol@#1\endcsname{#2}\fi}
\newcommand{\dcSecondOpt}[2][]{%
    \def\dc@argi{#1}\ifx\dc@argi\@empty\def\dc@iicontrol{#2}\else
    \expandafter\def\csname dc@iicontrol@#1\endcsname{#2}\fi}
```

The following are the default settings for these controls. The values for the macros `\digiDSWidth` and `\digiDSHeight` are defined in `\digiDisplaySpace`.

```
\dcFirstOpt{vcaption=b,hcaption=c,outerboxsep=0pt}
\dcSecondOpt{borderwidth=0bp,fboxsep=10bp,bordercolor=nocolor,bgop=.7}
```

`\useRollovers` `\noRollovers` Execute these commands to create rollovers for the long captions. The default is to use no rollovers.

```
\def\useRollovers{\def\dc@use@Rollover{*}%
    \def\hiddenPresets{\F{\FHidden}}}
\def\noRollovers{\let\dc@use@Rollover\@empty
    \def\hiddenPresets{}}
\let\dc@use@Rollover\@empty
```

Don't ask what this is.

```
\def\dc@fudge{\llap{.\hskip20in}}
```

`\longCapFmt` Use this command to apply a global format to the long captions. For example, `\longCapFmt{\bfseries\scriptsize}`. The default setting does nothing.

```
\newcommand{\longCapFmt}[1]{%
    \def\dc@longCapFmt{#1}}
\longCapFmt{}
```

`\dc@showPic` This is the command that places the large digital image in the display area.

```
\def\dc@showPic#1{\leavevmode\xBld{#1}\dc@fudge
    \vbox to0pt{\vss\hbox to0pt{\hss
```

The `inclgraphicx` of the `dc@commands` family has a secret macro named `\gc@incgfx@addkeys` inserted in its definition. By default, `\gc@incgfx@addkeys` is `\let` equal to `\@empty`. We change that definition here to include the `name` of the graphic, so the document author does not have to bother. We also scale the picture to fit in the display space.

```
    \def\dc@incgfx@addkeys{width=\digiDSWidth,%
        height=\digiDSHeight,keepaspectratio,name=#1}%
```

If there is a custom control for this image, we swap off the default one with the custom one.

```
    \@ifundefined{dc@icontrol@#1}{}{\expandafter\let\expandafter
    \dc@icontrol\expandafter=\csname dc@icontrol@#1\endcsname}%
```

If there is a custom control for this image for the second optional argument, we swap off the default one with the custom one.

```
    \@ifundefined{dc@iicontrol@#1}{}{\expandafter\let\expandafter
    \dc@iicontrol\expandafter=\csname dc@iicontrol@#1\endcsname}{}%
```

Finally, we are ready to execute the appropriate graphic caption command, with or without rollover.

```
    \expandafter\digiCap\dc@use@Rollover%
```

After determining which control for the first optional argument we insert `rollovername=ro#1` to give the rollover a pre-determined name that we know

and can use to give the rollover effect for the long caption, if requested. The rollover key is ignored, if `\digiCap*` is not used.

```
        [\dc@icontrol,rollovername=ro#1]{\null}[\dc@iicontrol]%
        {\dc@longCapFmt\csname#1Text\endcsname}%
\hss}\vss}\eBld}
```

`\digiDisplaySpace` A simple command to define a space to place the digital images into. The images are centered both horizontally and vertically in the display space. The first parameter is the height of the digital display, the second is the width. These dimensions are recorded in the macros `\digiDSHeight` and `\digiDSWidth` This command can be redefined, but the developer needs to define these two macros.

```
\newcommand{\digiDisplaySpace}[2]{%
    \def\digiDSHeight{#1}\def\digiDSWidth{#2}%
    \parbox[c][#1]{#2}{\centering\insertPhotos}%
}
```

`\insertPhotos` This is a user-interface to inserting the photos into the display area. Used by `\digiDisplaySpace`.

```
\newcommand{\insertPhotos}{\edef\dc@tmp@exp{\noexpand\@for
    \noexpand\@args:=\dc@presentationOrder}\dc@tmp@exp\do{%
        \edef\dc@tmp@exp{\noexpand\dc@showPic{\@args}}\dc@tmp@exp}%
}
```

`\shortCapFmt` User-interface to formatting the short captions.

```
\newcommand{\shortCapFmt}[1]{%
    \def\dc@showCaption##1##2{\makebox[0pt][c]{\xBld{##1}#1##2\eBld}}}
```

The default caption formatting is given below.

```
\shortCapFmt{\sffamily\bfseries\color{blue}}
```

`\insertCaptions` The main command for inserting captions, these can be placed above or below the display area.

```
\newcommand{\insertCaptions}{\dc@fudge\edef\dc@tmp@exp{\noexpand\@for
    \noexpand\@args:=\dc@presentationOrder}\dc@tmp@exp\do{%
        \edef\dc@tmp@exp{\noexpand\dc@showCaption{\@args}%
            {\noexpand\csname\@args Caption\noexpand\endcsname}}%
            \dc@tmp@exp}%
}
```

`\insertThumbs` `\setWidthOfThumbs` `\addvspacetorows` The command to insert the thumbs in a tabular environment. The first argument is the number of rows, and second argument is the number of columns.

```
\newcommand{\setWidthOfThumbs}[1]{{%
    \setlength{\dimen0}{#1}%
    \xdef\dc@thumbwidth{\the\dimen0}%
}}
\setWidthOfThumbs{0pt}
\newcommand{\addvspacetorows}[1]{\def\dc@addvspacetorows{#1}}
\addvspacetorows{1ex}%
```

```
\def\eq@tabSep{&}
\def\insertThumbs#1#2{\begingroup
    \count0=0\relax\count2=0\relax
    \def\dc@maxRows{#1}\def\dc@maxCols{#2}%
    \setlength{\dimen0}{\dc@thumbwidth}\ifdim\dimen0=0pt
    \setWidthOfThumbs{\linewidth/(\dc@maxCols)-\tabcolsep*2}\fi
    \edef\dc@thisArg{\dc@presentationOrder,}%
    \def\dc@insThumb@cr{\\[\dc@addvspacetorows]}%
    \kernOpt\begin{tabular}{*{#2}{m{\dc@thumbwidth}}}%
    \@whilenum\count0<#1\do{%
        \@whilenum\count2<#2\do{%
            \ifx\dc@thisArg\@empty
                \global\let\dc@insThumb@cr\relax
                \global\count0=\dc@maxRows
                \global\count2=\dc@maxCols\else
                \expandafter\dc@getNextArg\dc@thisArg\@nil
                \global\advance\count2by1\relax
                \ifx\dc@testArg\@empty
                \else\ifnum\count2=\dc@maxCols
                \else\eq@tabSep\fi\fi\fi
        }\dc@insThumb@cr
        \global\count2=0\relax\global\advance\count0by1\relax
    }%
\end{tabular}%
\endgroup}
\def\dc@getNextArg#1,#2\@nil{\dc@digi@thumbs{#1}\gdef\dc@thisArg{#2}}
```

3.4.3 Form fields and JavaScript

We define a push button with a normal, rollover and push appearance. The JavaScript actions makes the picture in the display space visible, and making the previous picture hidden.

The command, `\dc@digi@thumbs`, creates a push button with normal, down and rollover appearances. The JavaScript actions is to show execute the function `showThisPicture()`, which is defined as document JavaScript, below. The function manages the hiding and showing of layers, and if the `\useRollovers` is in effect, manages the rollover field created by `\digiCap*` command.

```
\def\normalAppr#1{n#1}
\def\downAppr#1{d#1}
\def\rolloverAppr#1{r#1}
\def\dc@digi@thumbs#1{\hfil%
{\dimen0=\widthOf{#1}bp\relax\dimen2=\heightOf{#1}bp\relax
 \ifdim\dimen0<\dimen2\relax
    \edef\dc@argi{\string!}\edef\dc@argii{\dc@thumbwidth}\else
    \edef\dc@argi{\dc@thumbwidth}\edef\dc@argii{\string!}\fi
    \xdef\dc@tmp@exp{\noexpand\resizebox{\dc@argi}{\dc@argii}}%
}\dc@tmp@exp{\pushButton[%
    \autoCenter{n}\BC{}\BG{}\S{S}\W0
    \A{\JS{%
```

```
        showThisPicture("#1");\r
        lastPicture="#1";\r
        this.dirty=false;
    }}\I{\normalAppr{#1}}\RI{\downAppr{#1}}\IX{\rolloverAppr{#1}}
    \TP{1}\FB{true}]{pb#1}{\widthOf{#1}bp}{\heightOf{#1}bp}}}
```

The JavaScript function manages the hiding and showing of layers, and if the `\useRollovers` is in effect, manages the rollover field. The argument `name` is the name of the graph to be shown. The name one to be hidden is saved as the value of `lastPicture`.

```
\begin{insDLJS}[showThisPicture]{digidjs1}{Show This Picture}
var lastPicture="";
function showThisPicture(name) {
    if (lastPicture != "") {
        var f = this.getField("dcRollover.ro"+lastPicture);
        if ( f != null ) f.display=display.hidden;
        toggleSetThisLayer(lastPicture,false);
    }
    var f = this.getField("dcRollover.ro"+name);
    if ( f != null ) f.display=display.visible;
    toggleSetThisLayer(name);
}
try { app.runtimeHighlight=false; app.focusRect=false; } catch(e) {};
\end{insDLJS}
⟨/digidisplay1⟩
```

4 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.