PCAD

Christian Paulsen

COLLABORATORS

	TITLE : PCAD	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Christian Paulsen	February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 PCAD **1**

1.1 Using the def2buttons compiler 1

1.2 About 'def2buttons' 1

1.3 The def2buttons compiler 2

1.4 The .def fileformat 3

Chapter 1

PCAD

1.1 Using the def2buttons compiler

```
                                                def2buttons

                  Copyright © 1994-96 Christian Paulsen


                 def2buttons is part of the PCAD package
                     PCAD is a shareware product!



Contents


                 About 'def2buttons'

                 How to 'compile' my own Tools-window

                 The .def description files
```

1.2 About 'def2buttons'

```
What is 'def2buttons'?


The def2buttons program may be used to create your own PCAD Tools-window.
def2buttons was a quick hack, rather than a comfortable interactive
GUI-builder.

PCAD uses the "pcad.icons" file, usually located in the "PCAD:" directory.
see 'Runtime Environment' for more details
This file is a binary description of all buttons in the Tools-window.
As "pcad.icons" is a static description, the Tools-window will be static for
one PCAD session, i.e. no buttons can be added, removed or changed while
running PCAD.
```

The "pcad.icons" file was created by using the def2buttons program. Each button is described by one .def file. def2buttons reads all the .def-files and creates a compact binary file, that is equivalent to "pcad.icons".

There are two kinds of buttons:

Std. action buttons

A std. action button is activated by clicking the icon (=button/gadget) with the left mousebutton and releasing it, while the mousepointer is still inside the icon (like every std. Amiga gadget). Then a single action is initiated. This action is a PCAD commandstring, which is sent to the PCAD command interpreter. As the commandstring is specified by each .def file, you can create your own button-menu, using any PCAD-commands, -scripts and/or Rexx macros.
e.g. the 'Zoom Out' button described by zoomout.def

Toggleselect buttons

These buttons behave like any std. Amiga toggleselect button. Activating the button will change its state from selected to unselected and vice versa. The .def file contents two commandstrings. One is sent to the command interpreter when the button switches to selected, the otherone when switching to unselected.
e.g. the 'Grid On/Off' button described by grid.def

1.3 The def2buttons compiler

How to "compile" my own Tools-window

The directory PCAD:icons holds all .def and .pic files that you need to compile the std. Tools-window. There are two kinds of files:

.def files

Each .def file defines one button (gadget,icon). It also contents the filename of the next button-description. By this, you get something like a linked list of files. All .def files together describe the Tools-window.

.pic files

The design (image) of each button is described by an IFF-ILBM file, that should be 26x22 (width x height) pixels big. The extension .pic is optional; you may use any names. The IFF filename is specified by the .def entry IMAGE=....

PCAD:def2buttons

The "compiler" needs one argument:

```
   The filename of the 1st (top-left) button.

It reads all following (NEXT=...) .def files and its .pic files, creates the
buttons and saves a binary image to 'test.picon'. This binary image uses the
same format as 'pcad.icons'. The 'test.picon' result may be used as
'pcad.icons' file. Don't forget to make a backup-copy of the original
'pcad.icons' file!

If you wan't to "recompile" the std. Tools-window you show follow these steps:
   - you are using an Amiga shell
   - enter the directory PCAD:icons
      >cd PCAD:icons

   - call the 'def2buttons' "compiler"
      >def2buttons zoomout.def

   - a window will be opened (your workbench should be big enough!)
     and the buttons are created.
         (This takes about 2s on my A4000/40/25MHz, reading the
         files from HD; i.e. the buttons won't pop up immediately.
         This may take some time on slow Amiga's. Watch your HD-LED
         [or Floppy-LED] to check if the program still works.)
     You may click the buttons to test them. The relating
     PCAD-command-sequence is written to the console.

   - close the window (by using the windows std. close-gadget)

   - if everything went well, the file 'test.picon' was created

   - backup your original 'pcad.icons' file
      e.g.
      >copy PCAD:pcad.icons PCAD:pcad.icons.bak

   - copy the 'test.picon' to 'PCAD:pcad.icons'
      >copy test.picon PCAD:pcad.icons

   - run PCAD to check the result
```

1.4 The .def fileformat

```
The .def button description

.def files are ASCII files that describe the PCAD Tools-window buttons.
The 1st line must always be 'FILETYPE=ICON', the second line defines an unique
name of the button (e.g. 'NAME=ZOOMOUT'). Then the position is set by
'LEFTEDGE=<integer>' and 'TOPEDGE=<integer>'. If the button should be
top-aligned to its (left) neighbour's top you should use the special
"<integer>" 'ALIGN' (e.g. 'TOPEDGE=ALIGN'). You may use the constant 'GAP' to
create the std. margin between blocks (e.g. 'TOPEDGE=GAP'). If the left-
and/or top-edge are given by relative values, you have to specify this by
'XRELATIVE=TRUE' and/or 'YRELATIVE=TRUE'. These values are related to the
right bottom of the previous button.

The buttons image is specified by 'IMAGE=<name>' (e.g.'IMAGE=zoomout.pic').
This should be an IFF ILBM picture, 26x22 pixels big.
```

```
A button may be of the 'TOGGLESELECT=TRUE' kind, like the grid-button of the
std. Tools-window. If it is a toggleselect button, there are some more
options:
   SELECTED=TRUE           # this sets the initial state;
   SELECT=<PCAD-cmd's>     # called when the state changes to selected
   DESELECT=<PCAD-cmd's>   # called when the state changes to unselected

These options are mutual exclusive with the std. 'ACTION=..' definition.
   ACTION=<PCAD-cmd's>     # called when button is clicked

See <PCAD-cmd's> for more details.

'ROOT=TRUE' should be specified by the 1st (i.e. top-left) button of
the Tools-window. Its used for the future, declaring hierarchical
button-menu trees.

The next button, is specified by the .def file of 'NEXT=..'
(e.g. 'NEXT=zoomin.def') building a linked list of button descriptions.

The '#' sign initiates a comment.

Examples:

'Zoom Out' button described by zoomout.def
'Grid On/Off' button described by grid.def

See other std. Tools-window definitions for more details.
```