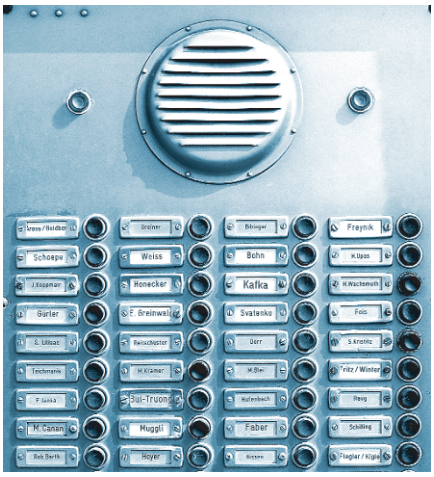
Pogo

Button Parade



Desktop accessories like the panel, icons and the start menu no longer hold the attraction they used to. Pogo adopts a completely new approach to launching programs at the click of a mouse, especially with regard to the optics.

BY ANDREA MÜLLER

If you have been looking for a flexible program launcher for a while now, *Pogo* (*http://www.ibiblio.org/propaganda/pogo/pogo-2.2.tar.gz*) may just put an end to that search. In contrast to a taskbar or icons, Pogo runs in a window of its own that can accommodate individual buttons.

You decide what functions the program offers. Whether it be a simple "switchbox" for launching programs, a system or mail monitor, there is very little that Pogo is not prepared to do for you.

Installing from Scratch

As binary packets are not currently available, you will have to compile Pogo yourself. To do so, you require *libpng*, *libtiff*, *libjpeg*, *imlib1*, *XFree*, and the accompanying development packages. Unzip the *pogo-2.2.tar.gz* archive file and change to the new directory this step creates, *pogo-2.2*.

The directory contains a pre-compiled version of the program, which is unlikely to run on your system. Remove this version by entering *make clean*, and compile Pogo by calling *make*. Then do *su -c "make install"* to install Pogo in */usr/local*; you need to know the *root* password to do this.

The command will also creates two links called *pogo* and *pogo-remote* in the */usr/bin* directory. Provided you keep the sourcecode directory, you can type *make clean* at any time to remove the software from your system; again you need to be *root* to do this.

Dress Rehearsal

Pogo's developer, *Bowie J. Poag*, supplies a few sample configurations to allow you to become accustomed to using Pogo. Simply type *pogo* in a terminal window to use them. The program creates a lot of output when launched.

If you try to be clever, and add an ampersand (&) to run the program in the background, you might well be driven to desperation by Pogo, as the program does not release the prompt. The workaround is to send the output to your system's "black hole" > */dev/null*:

```
pogo > /dev/null &
```

Pogo uses 64x64 pixel icons by default (see Figure 1). If you use low screen resolution, you can add the command line switch *-s* to modify the icon size. For example, *pogo -s 32* will tell Pogo to use 32x32 pixel icons.

Left-clicking a button launches the program specified in the */usr/local/pogo-2.2/configs/pogo.config* configuration file. Pogo displays a window title on mouse over to show you what program this is. The arrow icons at positions 1 and 3 allow you to switch between the four standard configuration files – the number on the button between them shows the current configuration file.

Mouse Driven

You can use the mouse to define the color scheme of the program launcher. Hold down the middle mouse button over a button, and move the mouse up or down to change the brightness. To change the color values of the button, move the mouse right or left instead. If the results are too garish for your likings, you can simply right click to remove the background color.

If you hold the right mouse button down, you can drag and drop the icon to another position. If your mouse has additional keys on the sides, Pogo uses these to vertically align icons, or save the colors and positions of the individual buttons. As normal users do not have write privileges for the */usr/local* directory, the latter function is fairly useless as long as you are working with a sample configuration.

If you use a more traditional mouse, you can use the keyboard to perform these tasks: [Tab] toggles between vertical and horizontal moves, and the space key saves your settings.

Tailor Made

To create an individual button bar, create a text file with a name of your choice in your home directory. You will need the file to store your Pogo menu: The first entry in the first line indicates whether

Figure 1: Pogo's first launch

Desktopia

Only you can decide how your desktop looks. With deskTOPia we regularly take you with us on a journey into the land of window managers and desktop environments, presenting the useful and the colorful viewers and pretty toys.

Listing 1: Sample Pogo Configuration

```
1 0 0
1 /usr/local/pogo-2.2/images/pogo-email2.jpg 7 62 180 sylpheed
2 /usr/local/pogo-2.2/images/pogo-mozilla.jpg 248 231 44 mozilla
3 /usr/local/pogo-2.2/images/pogo-console.jpg 235 3 30 xterm -e top
```

the bar should open vertically (*1*) or horizontally (*0*). You can then tell Pogo where it should place the bar (see Listing 1) using the top left corner of your screen as the point of origin for your desktop coordinate system.

All of the lines that follow contain definitions of the following type

```
ID icon color_in_RGB_notation ➥
program
```

for the individual icons. Replace *ID* with a unique number; a serial number is preferable. Replace *icon* with the name of the image file. This is the file that Pogo should display on the button. For this you might like to use one of the over 200 icons supplied that Pogo copies into */usr/local/pogo-2.2/images* during the installation phase.

Figure 2: Pogo with the sample configuration from Listing 1

This parameter is followed by the color of the button, which you can define by specifying the red, green and blue values, as in the combination for pure red, *255 0 0*, for example. If you prefer not to guess these values, you can use Gimp, *kcolorchooser* or another KDE graphics application to ascertain the correct values.

The line ends with the command the button will launch. Listing 1 shows a sample configuration that launches Pogo in the top left corner of the screen with three vertical button bars for the Sylpheed mail program, the Mozilla browser, and the system monitor *top* (see Figure 2). To tell the program to load the configuration when launched, simply pass the file to the program as a command line parameter *-c*, as in:

```
pogo -c /home/andi/mypogo
```

GLOSSARY

System load: *A value derived from the total number of processes, that are forced to wait for processor cycles, or hard disk access, within a period of time.*

Remote Control

pogo-remote, an additional program that allows you to send commands to an active Pogo session, makes Pogo really flexible. The remote control tool can perform a variety of tasks from loading configuration files, through moving individual buttons, to making icons flash. The syntax is as follows:

```
pogo-remote -d instruction -a ➥
value -b value -c value
```

The parameters *-a*, *-b* and *-c* are driven by the *instruction*. For example, *loadNewConfigFile*, which loads a new configuration file, expects the parameter *-c*, followed by the name of the configuration file.

```
pogo-remote -d loadNewConfig➥
File -c /usr/local/pogo-2.2/➥
configs/pogo-page-4.config
```

assigns the sample configuration number 4 to an active Pogo session. You can then issue the following command

```
pogo-remote -d strobeThisIcon➥
 -a 13 -b 20
```

to cause the inbox symbol (which has the ID*13* in the *pogo-page-4.config* configuration file) to flash 20 times (*-b 20*).

```
pogo-remote -d slideThisIcon➥
Left -a 13 -b 4
```

will move the mail symbol four places to the left (*-b 4*). The */usr/local/pogo-2.2/README* file describes the other parameters and their syntax.

Of course you can assign *pogo-remote* calls to Pogo buttons to save you typing those pesky command lines:

```
4 /usr/local/pogo-2.2/images/➥
pogo-2.jpg 255 229 136 ➥
pogo-remote -d loadNewConfig➥
File -c /home/andi/mypogo2
```

Well Scripted

A button that continually monitors the system load, and changes color to reflect the load, is another useful feature. You can load the *loadwatcher* script from the DVD to leverage this function. Then make the file executable and type

```
5 /usr/local/pogo-2.2/images/➥
pogo-wtf.jpg 255 229 136 ➥
/path/to/loadwatcher
```

to bind the script to the configuration file. The script reads the first value in the */proc/loadavg* file every ten seconds. If the value drops below 0.80, *pogo-remote* issues

```
pogo-remote -d colorizeThisIcon➥
 -a 5 -c 00ff33
```

to change the icon to green. If the system load increases, the icon turns red. The *-c* option also expects the RGB values of the color, but in hexadecimal format in this case, just like HTML files. If you want *loadwatch* to color the system monitor icon, which will be your first port of call in case of a system overload, replace *-a 5* with *-a 3*.

The *README* file for the program, and the scripts stored in the */usr/local/pogo-2.2/scripts* directory contain more useful information. If you enter your username, password and server address in the *checkmyemail.pogo* file located in that directory, the @ button shown in Figure 1 will monitor your POP mailbox and flash whenever you get mail. ■

THE AUTHOR

Andrea Müller is a law student who keeps herself busy with Linux. When time permits she likes to take a peek at other operating systems, such as QNX, BeOS and NetBSD.