



# Making a Startup-Splash for Perl Applications using Tk

By Trevelyn 2008  
[Douglas@WeakNetlabs.com](mailto:Douglas@WeakNetlabs.com)

## **ABSTRACT**

This shows the lab assistant how to easy create shell scripts, and Perl scripts that initiated by a sleek splash screen as in WeakNet Linux Version 1.0 and above. This method is not the most simplest way, but it's also not the longest. It's quick, fun, and I show examples of course!

## Part 1: coding

Firstly, create a Perl program. I will use this one:

```
#!/usr/bin/perl
print "hello world!\n";
```

And I'll call it perl.pl. Then create a Tk perl program that shows an image and has buttons. I will use this one:

```
#!/usr/bin/perl
use Tk;
my $main = new MainWindow(-title=>"WeakNet Labs 2009");
my $canvar = $main ->Canvas(-height=>"175", -width=>"422");
$main->geometry("424x218+300+300");
$canvar->grid(-columnspan=>"2");
my $file = 'demos/images/perlpic.gif';
my $img =
    $canvar->Photo( 'IMG',
        -file => Tk->findINC($file) );
    $canvar->create( 'image',0,0,
        '-anchor' => 'nw',
        '-image' => $img );
$main->Button(-text=>"    Begin    ", -command=>sub{&start;})->grid(
$main->Button(-text=>"    Exit    ", -command=>sub{exit;}),
-pady=>"5");

MainLoop;
sub start { system ("perl.pl &");exit; }
```

I will call this one perlstart.pl

### Tiny breakdown of the code:

The first line calls the interpreter: Perl. The second tells Perl to use the Tk module for graphics. The third says I want to make a main window to show everything in. The fourth says to create a Canvas via the Tk module to display the splash image in. The fifth line, says that I want the main window to be 424 pixels by 218 pixels and to be located at pixel 300 (y) , pixel 300 (x) coordinates from the top left of the Desktop. The sixth line states that I want to span the image in the Canvas for two rows. This is so I can put two buttons at the bottom and line them up neatly. The next line says that I want to use a certain image, and then I declare the image \$img variable with my canvas using the “Photo” function of the Tk Module.

I want the image to have no top and left borders “0,0,” and I want to “anchor” the image to the northwest part of the window (to size the window easily once finished). Now I declare my buttons, Begin and Exit that will be in the two rows below the image in the canvas. The commands will be “GO TO subroutine 'start’” and “exit” respectively. Then I pad the buttons so they don't look squished

against the image in the grid() function of Tk with “-pady=>5” and then finish the MainLoop. Then I declare the “start” subroutine and what it does: run the perl.sh script I still need to write.

Now, we want to write a bash script to point to the perl.pl script I wrote above. I will explain why all this is necessary when I'm through with getting you a functional program called with the “begin” button of the splash. I will use the script below:

```
#!/bin/bash
gnome-terminal --geometry="80x40" -x bash -c "perl /usr/bin/perl.pl;bash"
```

Okay, that was easy eh? I simply create a script that runs “perl /usr/bin/perl.pl” in gnome-terminal.

### **How this all comes together.**

In Gnome, you can easily customize the menu bar by right clicking on the “Applications” title and select “Edit Menus” To add your new application you need the files all to point to one another and make them executable. Here's how to do that:

Say I want them in my \$PATH. I can echo my \$PATH and find a convenient folder to put them in such as /usr/bin.

```
# sudo chmod a+rx perl.pl && chmod a+rx perlstart.pl && chmod a+rx perl.sh
# sudo cp perl.pl /usr/bin && cp perl_start.pl /usr/bin && cp perl.sh
/usr/bin
```

There we go, all three in my \$PATH and executable! Now If I run perlstart.pl It will open the GUI Tk application that has the splash and two buttons. That in turn will open the perl.sh shell file if and only if “Begin” is clicked on.

Now we need to put the splash image in the right spot. FindINC() function's root directory seems to be in “/usr/lib/perl5/” (for basic GNU Debian) and if you look at the code above, it goes even deeper to “demos/images/” Put the image there. If you have a basic version of Tk, you will have to make the image a GIF format file. Otherwise you can get the png modules or whatever installed.

Then the splash image will exit, as told in this line:

```
sub start { system ("perl.pl &");exit; }
```

## The Easy Way

There's another way to place an image into a program without using a canvas. You can simply use the “Label” function and the “-image” argument. Here is an example:

```
#!/usr/bin/perl
use Tk;
my $main = MainWindow->new(-title=>"Netgh0st Version 3.0 2008 WeakNet Labs");
$main->geometry("400x250+150+200");

my $img    = $main->Photo(-file=>'/usr/lib/perl5/Tk/demos/images/earth.gif');
my $beginb = $main->Photo(-file=>'/usr/lib/perl5/Tk/demos/images/start-
button.gif');
my $end    = $main->Photo(-file=>'/usr/lib/perl5/Tk/demos/images/exit-
button.gif');

$main->Label(-image=>$img)->grid(-row=>"0", -columnspan=>"2");

$main->Button(-image=>$beginb, -command=>sub{&begin; exit;})->grid(-row=>"2",
-column=>"0", -pady=>"5");
$main->Button(-image=>$end, -command=>sub{exit; })->grid(-row=>"2",
-column=>"1", -pady=>"5");
MainLoop;
sub begin {
    system ("ngh0st_link &");exit;
}
```

Here you see I declare the images using the “Photo” function of Tk. \$img, \$beginb, and \$end will show up in the windows were I place them with the “Label” or “Button” functions below the declarations. In this easy way, you don't necessarily have to have your images in the default Tk, folder for “demos/images/” as before. I keep them there to be consistent.

## Conclusion

Before I quit typing, I wanna take this chance to tell you why 3 files are needed. Well, the splash Tk application with the two buttons, seems to hang when the child process (gnome-terminal in our case) has opened and will not close until the child process is killed or complete. I'm not %100 sure why this is, but have tested everything to my knowledge to make this a smaller process. You could make this process smaller, you could have Perl create the shell script and the second Perl application itself once started and delete it upon completion.

But, this is an introduction. Something to get your feet wet with before jumping right in. Learning Tk was harder for me as I learned basic Perl first and had to re-learn syntax I felt like I should have had an easy grasp on already. Tackle it with an open mind. Have fun with it. Programming is so rewarding and fun (when it works!)

Here is a screen shot of one of my splashes I used in **WeakNet Linux Assistant version 1.0**



figure 0: The splash start for Netgh0st version 3.0 (the white border is the "Canvas()" function you see in the above code. If your image displays only partially, or too small in the white border you can change it's size with '-width=>"n", -height=>"n"' parameters in the Canvas() function.