How Perl Brought The Funny

Mr. Voice, Perl/Tk, and ComedyWorx
H. Wade Minter
minter@lunenburg.org
http://www.lunenburg.org/mrvoice/
About Me

• Name: H. Wade Minter
• Originally from Virginia, graduate of William & Mary
• Unix sysadmin and geek-of-all-trades.
• Perl “street cred? Not much.
• Currently working for a consulting company on a contract with a soul-sucking State Government project.
Background

- ComedyWorx used to run its sound system off of tapes and CDs.
- Tapes are prone to breaking.
- CDs are slow when switching tracks.
- Both are hard to find in a stack of media.
My idea - digital audio

- Quick access.
- Easy backup.
- Mmm ... geeky.
How It’s Grown!
Why Perl/Tk?

- Already comfortable with Perl.
- Consensus good choice.
- Excellent documentation.
- Happened to see a book in the bookstore.
- Other Options: wxPerl, Gtk-Perl, PerlQt.
Things On The Agenda

- Layout management
- Buttons as an example widget.
- Keyboard bindings.
- Cross-platform coding.
- Your questions (hopefully).
Useful Information
Widget Sampler

- Labels
- Buttons
- Radiobuttons
- Checkbuttons
- Listboxes
- Text Entry
- Frames
Layout Managers

• Choices: Pack, Place, and Grid
• Pack: The most common one. Uses the “allocation rectangle” system.
• Place: Pixel-level placement.
• Grid: Spreadsheet-like cells
• I chose Pack.
Pack

• Allocation rectangle system somewhat tough to grasp.

• Each widget you pack() has a virtual rectangle that extends horizontally or vertically across its containing widget.

• Once you get it, it’s very powerful.
$mw = MainWindow;
pack(-side=>'top');
pack(-side=>'left');
pack(-side=>'bottom');
pack(-side=>'right');
Advantages

• Very fluid layout.
• Easy to resize.
• How to line up multiple items (buttons, checkboxes, etc.)?
Frames

- Frame widget is simply a container for other widgets.

- `Pack()` a frame, then pack other widgets inside the frame.

- Let’s look at some example code:
This Chart Means Nothing
my $statusframe = $mw->Frame()->pack(
    -side => 'bottom',
    -anchor => 's',
    -fill => 'x'
);

our $playbutton = $statusframe->Button(
    -text => "Play Now",
    -command => [ \&play_mp3, $mainbox ]
)->pack( -side => 'left' );

our $stopbutton = $statusframe->Button(
    -text => "Stop Now",
    -command => \&stop_mp3
)->pack( -side => 'right' );

$statusframe->Label(
    -textvariable => \$status,
    -relief => 'sunken'
)->pack(
    -anchor => 'center',
    -expand => 1,
    -padx => 5,
    -fill => 'x'
);
Welcome to Mr. Voice version 1.10.7
Widgets
Mr. Voice - Main Window
Buttons

- Perl/Tk is event driven
- Clicky clicky do something
- Buttons are a familiar interface.
- Click on a button and execute a callback.
$summarybox->Button(
    -text => "Close",
    -command => sub {
        $summarybox->destroy if Tk::Exists($summarybox);
    }
)->pack();
• Inline anonymous subroutine

• When the button is pressed, it runs the subroutine, then goes back to waiting.

• The application (usually) blocks while the subroutine is running.

• You can also call named subroutines.
my $playlistbutton = $holdingtank->Button(
    -text => "Launch Playlist",
    -command => \&launch_tank_playlist
  )->pack();
sub launch_tank_playlist {
    # Launch an m3u playlist from the contents of the holding tank
    my @indices = $tankbox->get( 0, 'end' );
    return if ( $#indices < 0 );
    my ( $fh, $filename ) = tempfile( SUFFIX => '.m3u', UNLINK => 1 );
    print $fh "#EXTM3U\n";
    foreach my $item (@indices) {
        my ( $id, $description ) = split( /:/, $item );
        my $file = get_info_from_id($id)->{filename};
        my $path = catfile( $config{filepath}, $file );
        print $fh "$path\n";
    }
    close($fh) or die;
    system("$config{'mp3player'} $filename");
}
• That subroutine just ran and exited.

• Can, of course, call subroutines with arguments.

• Can use an image as a button.
$holdingtank->Button(
    -image => $arrowup,
    -command => [ \&move_tank, "-1"
)
)->pack( -side => 'left' );
sub move_tank
{

# Blatantly cribbed from Mastering Perl/Tk's Tk::NavListbox example
my $lb    = $tankbox;
my $direction = shift;
my (@selection) = $lb->curselection;
my $index     = $selection[0];

# Sanity checks
return if ( $index == 0 && $direction == -1 );
return if ( $index == $lb->size() - 1 && $direction == 1 );

my $newindex = $index + $direction;

my $item = $lb->get($index);
$lb->delete($index);
$lb->insert( $newindex, $item );
$lb->selectionSet($newindex);
}

Binding Keys
Binding Keys

• Can map keyboard actions to events in the GUI.

• Mr.Voice uses hotkeys to play songs, as well as a few other key combos.

• Use the bind() method.
$mainbox->bind( "<Double-Button-1>",
\&play_mp3 );

$mw->bind( "<Control-Key-p>",
[ \&play_mp3, "Current" ] );
• First bind means that if you double-click in the $mainbox (Tk::Listbox), it runs the play_mp3() subroutine.

• Second bind means that if you press Ctrl-P anywhere in the main window ($mw), it will run the play_mp3() subroutine with an argument of “Current”

• Both binds pass a reference to their calling widget as the first argument.
Cross-Platform Coding

• The Tk widget set doesn’t look quite like native Win32, or quite like GTK/Qt, but is fairly close to both.

• Perl/Tk provides some functions (getOpenFile, getSaveFile) that use native widgets under Win32

• No native OS X port (yet).
Mr.Voice - Linux
Mr. Voice - Win32
Cross-Platform Coding

- Use the $^O variable for OS-specific cases.
- Use File::Spec for path issues.
- Use Win32::GetShortPathName to avoid problems.
- That’s about it, really.
# Check to see if we're on Windows or Linux, and set the RC file accordingly.

if ( $^O eq "MSWin32" ) {
  $rcfile = "C:\\mrvoice.cfg";
  my $logfile = "C:\\mrvoice.log";
  open( STDOUT, ">$logfile" );
  open( STDERR, ">&STDOUT" );
}
else {
  our $homedir = "~";
  $homedir =~ s{ ^ ~ ( [^/]*) }{ $1 ? (getpwnam($1))[7]
    : ( $ENV{HOME} || $ENV{LOGDIR} || (getpwuid($>))[7]
      || (getpwpid($>))[7] )
  }ex;
  $rcfile = "$homedir/.mrvoice.rc";
  my $logfile = "$homedir/mrvoice.log";
  open( STDOUT, ">$logfile" );
  open( STDERR, ">&STDOUT" );
}
Resources
Your Best Friend
Other Resources

• *The Perl Journal* - April 2004

• Usenet: comp.lang.perl.tk (*Extremely helpful!*)

• Mailing List: ptk@lists.Stanford.EDU (email majordomo@lists.stanford.edu with “subscribe ptk”)

• WWW: http://www.perltk.org/ (fairly crap-tastic)
Thanks & Info

• Mr. Voice: http://www.lunenburg.org/mrvoice/
• Wade: minter@lunenburg.org
• Vote for me: http://www.minterforhouse.org/
</plug>
PGP Keysigning?

- pub 1024D/24460EC7 1998-08-20
  Wade Minter
  <minter@lunenburg.org>
- sub 2048g/D43AD322 1998-08-20
- Primary key fingerprint: 8F2B E420 9C05 05FC 5149 08C7 A380 F0B3 2446 0EC7