

TIP #159: Extending Tk 'wm' Command to Support Coloured Icons

Source: <http://coding.derkeiler.com/Archive/Tcl/comp.lang.tcl/2003-10/0103.html>

From: Georgios Petasis (petasis_at_iit.demokritos.gr)

Date: 10/02/03

Date: Thu, 2 Oct 2003 15:59:50 +0000 (UTC)

TIP #159: EXTENDING TK 'WM' COMMAND TO SUPPORT COLOURED ICONS

Version: \$Revision: 1.1 \$

Author: Georgios Petasis <petasis_at_iit.demokritos.gr>

State: Draft

Type: Project

Tcl-Version: 8.5

Vote: Pending

Created: Wednesday, 01 October 2003

URL: <http://purl.org/tcl/tip/159.html>

WebEdit: <http://purl.org/tcl/tip/edit/159>

Post-History:

ABSTRACT

Currently, Tk lacks a mechanism for allowing scripts to place colour icons in the window manager decorations of a toplevel window. Tk supports only the placement of monochrome bitmaps through the `_wm iconbitmap_` and `_wm iconmask_` commands. This TIP proposes an extension of the `_wm_` command with the `_iconphoto_` subcommand, which will pass a set of photo images as possible window manager icons.

RATIONALE

Almost all modern window managers or desktop environments offer support for using colour icons in toplevel window decorations. Tk has made some steps in this direction by allowing `_wm iconbitmap_` command to also accept a Windows icon file, to be used as an icon for a toplevel under windows. This solution is quite incomplete, and of course works only under windows. No support for colour icons under Unix is currently offered and I (think) the same is also true for Mac OS.

comp.lang.tcl: TIP #159: Extending Tk 'wm' Command to Support Coloured Icons

This TIP proposes the introduction of a new subcommand to the `_wm_` command, the `_iconphoto_` subcommand. (`_iconimage_` was an alternative that I rejected because bitmaps are also considered images by Tk). This subcommand will accept a Tk window name and a set of photo image names. These images will then be passed to the window manager as possible window icons.

The `_wm iconphoto_` subcommand will be available under all platforms. However, at this stage of development I will focus mainly on Unix: I don't know anything about Mac OS (so others must help in this direction) and support for windows already exists. The final goal will be to support at least windows and Unix. There is currently a patch that implements the Unix port of `_wm iconphoto_`.

Currently, two different approaches are used by Unix applications to specify colour icons. The first approach uses the same mechanism as `_wm iconbitmap_` to pass the icon: instead of passing a monochrome pixmap (that is a bitmap), they pass a colour pixmap with 3 planes (no transparency). This approach works with many modern window managers (including the ones used by KDE and GNOME). However, as Joe English noted this approach violates the ICCCM, despite the fact that it is used by some applications (i.e. `gvim`).

The second approach (again pointed by Joe English) is the one proposed by `freedesktop.org`: The application defines a window property that holds a special encoding of the colour icons. I don't know exactly how many window managers follow this proposal, but the window managers in two popular desktops (KDE and GNOME) support it. My proposal is of course to follow the second approach, as it does not violate the ICCCM.

The window property that should be defined is `"_NET_WM_ICON"`. From the specifications at <http://www.freedesktop.org>:

```
_NET_WM_ICON
```

```
_NET_WM_ICON CARDINAL[][2+n]/32
```

This is an array of possible icons for the client. This specification does not stipulate what size these icons should be, but individual desktop environments or toolkits may do so. The Window Manager MAY scale any of these icons to an appropriate size.

This is an array of 32bit packed CARDINAL ARGB with high byte being A, low byte being B. The first two cardinals are width, height. Data is in rows, left to right and top to bottom.

Currently both KDE and GNOME window managers scale the provided icons to proper sizes.

SPECIFICATION

=====

This document proposes the following changes to the Tk core:

1. The addition of a new subcommand (`_iconphoto_`) to the `_wm_` command. This subcommand will accept the following arguments:

 window: the name of a Tk toplevel window.

 photo_image ?photo_image2 ...?:
 a set of Tk photo image names, that will be send to the window manager as potential icons.

Currently, no other facilities are planned (like for example getting back the image names that are currently used as icons).

Also, no facilities are provided to request the window manager what icon sizes are preferred, although Xlib offers a relevant function.

REFERENCE IMPLEMENTATION

=====

Here is a possible implementation for the Unix platform. The patch is very small, so I have included it here:

```
--- tk_XP/unix/tkUnixWm.c 2003-09-02 14:56:46.000000000 +0300
+++ tk/unix/tkUnixWm.c 2003-10-01 10:04:40.000000000 +0300
@@ -395,6 +395,9 @@
 static int WmIconnameCmd _ANSI_ARGS_((Tk_Window tkwin,
                                     TkWindow *winPtr, Tcl_Interp *interp, int objc,
                                     Tcl_Obj *CONST objv[]));
+static int WmIconphotoCmd _ANSI_ARGS_((Tk_Window tkwin,
+ TkWindow *winPtr, Tcl_Interp *interp, int objc,
+ Tcl_Obj *CONST objv[]));
 static int WmIconpositionCmd _ANSI_ARGS_((Tk_Window tkwin,
                                     TkWindow *winPtr, Tcl_Interp *interp, int objc,
                                     Tcl_Obj *CONST objv[]));
@@ -971,7 +974,8 @@
     "aspect", "attributes", "client", "colormapwindows",
     "command", "deiconify", "focusmodel", "frame",
     "geometry", "grid", "group", "iconbitmap",
- "iconify", "iconmask", "iconname", "iconposition",
+ "iconify", "iconmask", "iconname",
+ "iconphoto", "iconposition",
     "iconwindow", "maxsize", "minsize", "overrideredirect",
     "positionfrom", "protocol", "resizable", "sizefrom",
     "stackorder", "state", "title", "transient",
@@ -980,7 +984,8 @@
     WMOPT_ASPECT, WMOPT_ATTRIBUTES, WMOPT_CLIENT,
     WMOPT_COLORMAPWINDOWS,
```

comp.lang.tcl: TIP #159: Extending Tk 'wm' Command to Support Coloured Icons

```
WMOPT_COMMAND, WMOPT_DEICONIFY, WMOPT_FOCUSMODEL, WMOPT_FRAME,
WMOPT_GEOMETRY, WMOPT_GRID, WMOPT_GROUP, WMOPT_ICONBITMAP,
- WMOPT_ICONIFY, WMOPT_ICONMASK, WMOPT_ICONNAME, WMOPT_ICONPOSITION,
+ WMOPT_ICONIFY, WMOPT_ICONMASK, WMOPT_ICONNAME,
+ WMOPT_ICONPHOTO, WMOPT_ICONPOSITION,
WMOPT_ICONWINDOW, WMOPT_MAXSIZE, WMOPT_MINSIZE,
WMOPT_OVERRIDEREDIRECT,
WMOPT_POSITIONFROM, WMOPT_PROTOCOL, WMOPT_RESIZABLE,
WMOPT_SIZEFROM,
WMOPT_STACKORDER, WMOPT_STATE, WMOPT_TITLE, WMOPT_TRANSIENT,
@@ -1072,6 +1077,8 @@
return WmIconmaskCmd(tkwin, winPtr, interp, objc, objv);
case WMOPT_ICONNAME:
return WmIconnameCmd(tkwin, winPtr, interp, objc, objv);
+ case WMOPT_ICONPHOTO:
+ return WmIconphotoCmd(tkwin, winPtr, interp, objc, objv);
case WMOPT_ICONPOSITION:
return WmIconpositionCmd(tkwin, winPtr, interp, objc, objv);
case WMOPT_ICONWINDOW:
@@ -2061,6 +2068,122 @@
/*
*-----
*
+ * WmIconphotoCmd --
+ *
+ * This procedure is invoked to process the "wm iconphoto"
+ * Tcl command.
+ * See the user documentation for details on what it does.
+ *
+ * Results:
+ * A standard Tcl result.
+ *
+ * Side effects:
+ * See the user documentation.
+ *
+ *-----
+ */
+
+static int
+WmIconphotoCmd(tkwin, winPtr, interp, objc, objv)
+ Tk_Window tkwin; /* Main window of the application. */
+ Tk_Window *winPtr; /* Toplevel to work with */
+ Tcl_Interp *interp; /* Current interpreter. */
+ int objc; /* Number of arguments. */
+ Tcl_Obj *CONST objv[]; /* Argument objects. */
+ {
+ register WmInfo *wmPtr = winPtr->wmInfoPtr;
+ Tk_PhotoHandle photo;
+ Tk_PhotoImageBlock block;
+ int i, size = 0, width, height, index = 0, x, y;
+ long R, G, B, A;
```

comp.lang.tcl: TIP #159: Extending Tk 'wm' Command to Support Coloured Icons

```
+ long *iconPropertyData;
+ unsigned char *pixelByte;
+
+ if (objc < 4) {
+ Tcl_WrongNumArgs(interp, 2, objv,
+ "window photo_image ?photo_image2 ...?");
+ return TCL_ERROR;
+ }
+ /*
+ * Iterate over all images to retrieve their sizes, in order to allocate a
+ * buffer large enough to hold all images.
+ */
+ for (i = 3; i < objc; i++) {
+ photo = Tk_FindPhoto(interp, Tcl_GetString(objv[i]));
+ if (photo == NULL) return TCL_ERROR;
+ Tk_PhotoGetSize(photo, &width, &height);
+ /* We need to cardinals for width & height and one cardinal for each
+ * image pixel. */
+ size += 2 + width * height;
+ }
+ /* We have calculated the size of the data. Try to allocate the needed
+ * memory space. */
+ iconPropertyData = (long *) Tcl_AttemptAlloc(sizeof(long)*size);
+ if (iconPropertyData == NULL) {
+ return TCL_ERROR;
+ }
+ memset(iconPropertyData, 0, sizeof(long)*size);
+
+ for (i = 3; i < objc; i++) {
+ photo = Tk_FindPhoto(interp, Tcl_GetString(objv[i]));
+ if (photo == NULL) {
+ Tcl_Free((char *) iconPropertyData);
+ return TCL_ERROR;
+ }
+ Tk_PhotoGetSize(photo, &width, &height);
+ Tk_PhotoGetImage(photo, &block);
+ /*
+ * Each image data will be placed as an array of 32bit packed
+ * CARDINAL, in a window property named "_NET_WM_ICON":
+ * _NET_WM_ICON
+ *
+ * _NET_WM_ICON CARDINAL[][2+n]/32
+ *
+ * This is an array of possible icons for the client.
+ * This specification does not stipulate what size these icons should
+ * be, but individual desktop environments or toolkits may do so.
+ * The Window Manager MAY scale any of these icons to an appropriate
+ * size.
+ *
+ * This is an array of 32bit packed CARDINAL ARGB with high byte being
+ * A, low byte being B. The first two cardinals are width, height.
```

comp.lang.tcl: TIP #159: Extending Tk 'wm' Command to Support Coloured Icons

```
+ * Data is in rows, left to right and top to bottom.
+ */
+
+ /*
+ * Encode the image data in the iconPropertyData array.
+ */
+ iconPropertyData[index++] = width;
+ iconPropertyData[index++] = height;
+ for (y = 0; y < height; y++) {
+ for (x = 0; x < width; x++) {
+ R = *(block.pixelPtr + x*block.pixelSize +
+ y*block.pitch + block.offset[0]);
+ G = *(block.pixelPtr + x*block.pixelSize +
+ y*block.pitch + block.offset[1]);
+ B = *(block.pixelPtr + x*block.pixelSize +
+ y*block.pitch + block.offset[2]);
+ A = *(block.pixelPtr + x*block.pixelSize +
+ y*block.pitch + block.offset[3]);
+ pixelByte = (unsigned char *) &iconPropertyData[index];
+ pixelByte[3] = A;
+ pixelByte[2] = R;
+ pixelByte[1] = G;
+ pixelByte[0] = B;
+ index++;
+ }
+ }
+ }
+ XChangeProperty(winPtr->display, wmPtr->wrapperPtr->window,
+ Tk_InternAtom(tkwin, "_NET_WM_ICON"),
+ XA_CARDINAL, 32, PropModeReplace,
+ (unsigned char *) iconPropertyData, size);
+ Tcl_Free((char *) iconPropertyData);
+ return TCL_OK;
+ }
+ ^L
+ /*
+ * -----
+ *
+ * WmIconpositionCmd --
+ *
+ * This procedure is invoked to process the "wm iconposition"
```

Note that also I have a reference implementation of the first approach (passing a colour pixmap instead of a bitmap), in case we decide to follow this approach (or both simultaneously :-).

NOTES

=====

I don't know if I have handled the assignment of the four bytes inside the long correctly. Currently the code works fine under Linux. It would

comp.lang.tcl: TIP #159: Extending Tk 'wm' Command to Support Coloured Icons

be great if somebody could re-write the assignment to use bit-shift operations :-)

COPYRIGHT

=====

This document has been placed in the public domain.

TIP AutoGenerator – written by Donal K. Fellows

[[Send Tcl/Tk announcements to tcl-announce@mitchell.org
Announcements archived at http://groups.yahoo.com/group/tcl_announce/
Send administrivia to tcl-announce-request@mitchell.org
Tcl/Tk at <http://tcl.tk/>]]