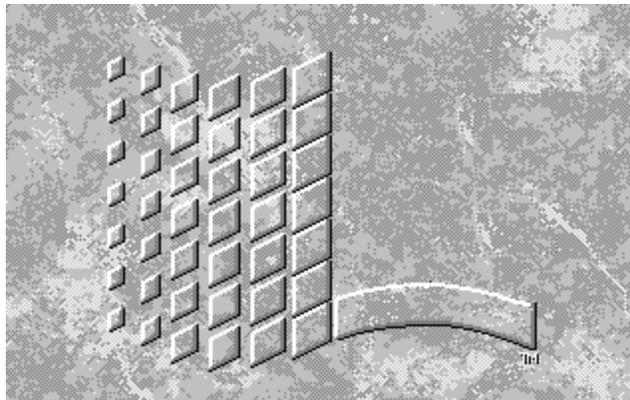
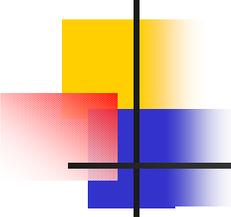


Win32 Port of ROOT GUI

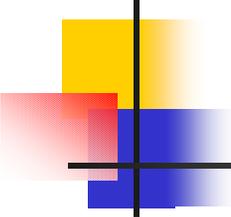


Bertrand Bellenot - bertrand.bellenot@alusuisse.com



Features

- Same environment on every system :
 - Same look and feel on every platform.
- Simplify the code maintenance :
 - No need to care about a « windows specific code ».
- Simplify functionality extension :
 - No need to implement the code twice, once for windows and once for other OS.
 - Only use TVirtualX.



Actual Status

- The actual code uses a modified version of gdk and glib, the GIMP low-level libraries ported on win32. In practice, this means that we only need to link with gdk.lib, glib.lib and iconv.dll as additional libraries (hopefully less in the future). These libraries are under LGPL, so there are no licensing issues in using and distributing them.
- As original version of gdk was not doing everything needed by root (as font orientation!), I did have to slightly modify the original code.

Example of Win32 implementation from TGX11 with gdk - most favorable case !

Here is the X11 code :

```
// _____  
void TGX11::MoveResizeWindow(Window_t id, Int_t x, Int_t y,  
                             UInt_t w, UInt_t h)  
{  
    // Move and resize a window.  
  
    XMoveResizeWindow(fDisplay, (Window) id, x, y, w, h);  
}
```

And here is the win32 one :

```
// _____  
void TGWin32::MoveResizeWindow(Window_t id, Int_t x, Int_t y,  
                               UInt_t w, UInt_t h)  
{  
    // Move and resize a window.  
  
    gdk_window_move_resize((GdkWindow *) id, x, y, w, h);  
}
```

Example of Win32 implementation from TGX11 - less favorable case...

Here is the X11 code :

```
//_____
void TGX11::TranslateCoordinates(Window_t src, Window_t dest, Int_t src_x,
                                Int_t src_y, Int_t &dest_x, Int_t &dest_y, Window_t &child)
{
    Window xchild;

    XTranslateCoordinates(fDisplay, (Window) src, (Window) dest, src_x,
                        src_y, &dest_x, &dest_y, &xchild);
    child = (Window_t) xchild;
}
```

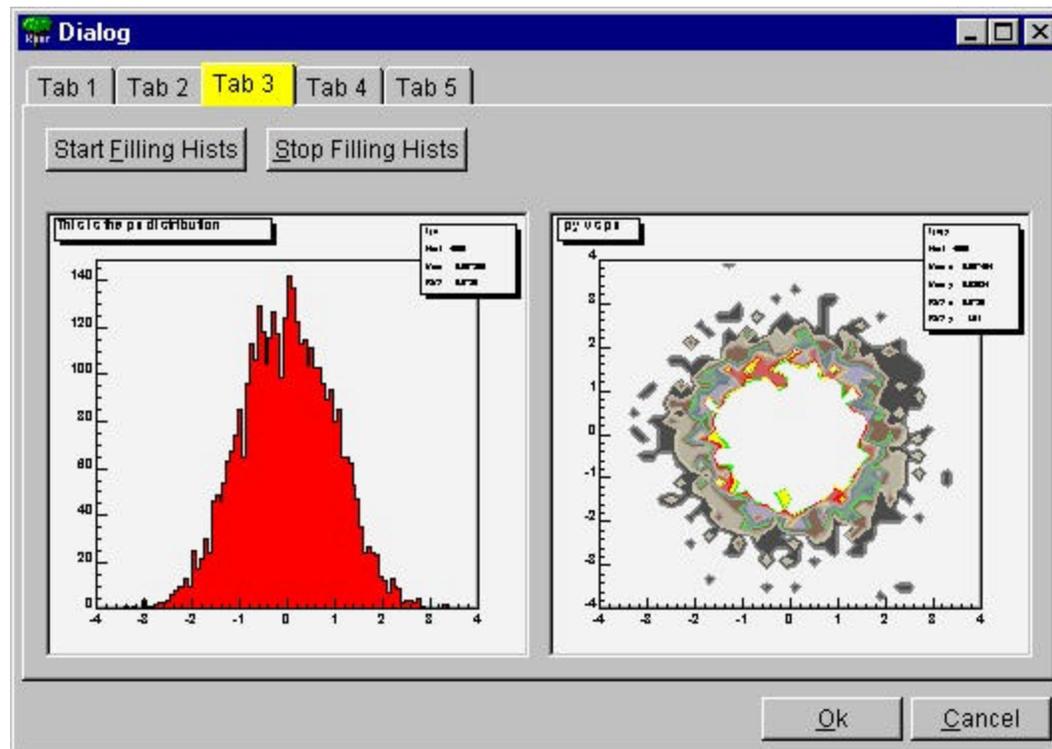
Example of Win32 implementation from TGX11 - less favorable case... (part 2)

And here is the Win32 one :

```
// _____  
void TGWin32::TranslateCoordinates(Window_t src, Window_t dest, Int_t src_x,  
    Int_t src_y, Int_t &dest_x, Int_t &dest_y, Window_t &child)  
{  
    HWND sw, dw, ch = NULL;  
    POINT point;  
    sw = (HWND)GDK_DRAWABLE_XID ((GdkWindow *)src);  
    dw = (HWND)GDK_DRAWABLE_XID ((GdkWindow *)dest);  
    point.x=src_x;  
    point.y=src_y;  
    MapWindowPoints(  
        sw,          // handle of window to be mapped from  
        dw,          // handle to window to be mapped to  
        &point,      // pointer to array with points to map  
        1);          // number of structures in array  
    ch=ChildWindowFromPoint(dw,point);  
    child=(Window_t) gdk_xid_table_lookup (ch);  
    if(child == src) child = (Window_t) 0;  
    dest_x = point.x;  
    dest_y = point.y;  
}
```

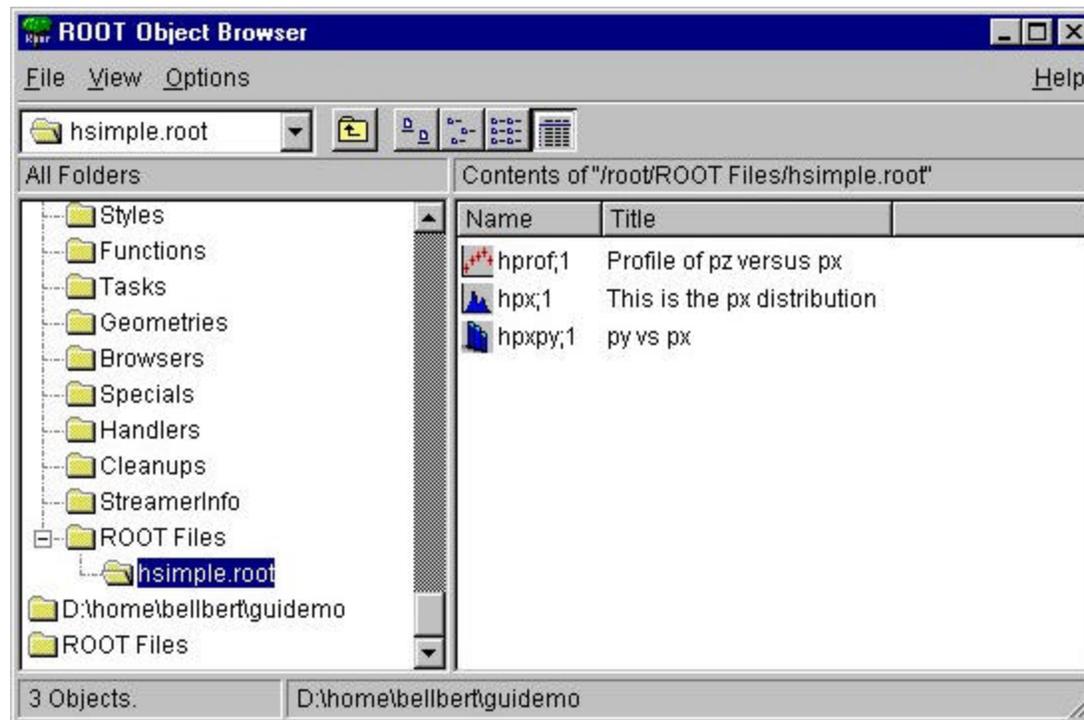
Actual Status

- Dialog screenshot (from guitest)



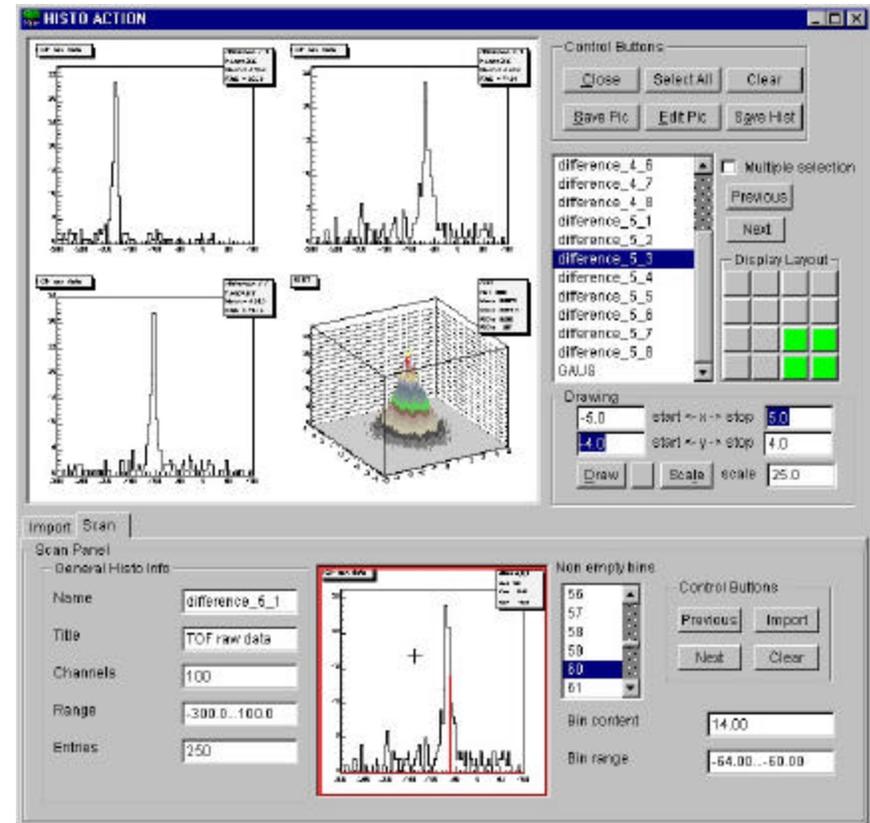
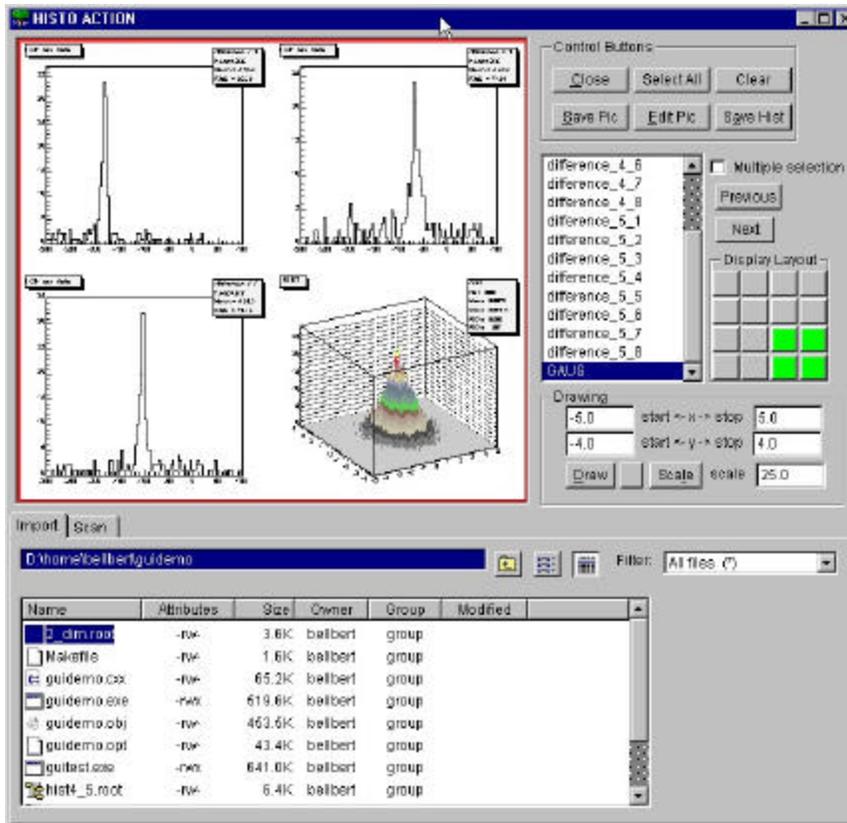
Actual Status

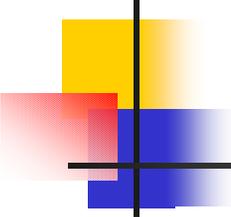
- Browser screenshot



Actual Status

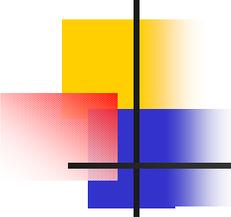
- A more complex GUI application (thank's to Dmitri Vassili).





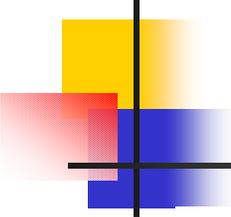
Threads problem

- On Windows NT, we need to have threads, and there is at least one thread per process, so if we need to share events, messages, or other resources into the same process, we need to create different threads.
- For interactive root, we need at least two threads : one to handle console events and another one to handle GUI events. This means that actually the win32 port works only in a standalone application and not into CINT, because console AND GUI message handling are into the same main thread (process) :



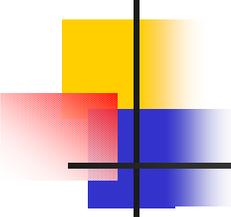
Threads problem

- Into TWinNTSystem::DispatchOneEvent() :
 gROOT->GetApplication()->HandleTermInput();
 if (gXDisplay && gXDisplay->Notify())
 if (!pendingOnly) return;
- In fact, we can start interactive root (root.exe), but the process will wait for console event before to handle GUI events. On the contrary, a standalone GUI application will work because there is no need to handle console events.



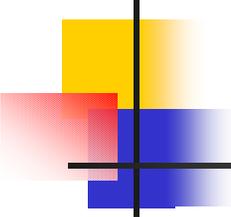
Other points

- Some characters are not displayed (like √ into symbol font). It is a problem with gdk, characters greater than 0x080 are not displayed (UTF-8 coding). I will solve this into gdk itself.
- Some problems with icon 's transparency.
- The event handling is not perfect : mouse clicks come twice in some cases – maybe it comes from gdk, to check – not really a big issue.
- OpenGL does not work yet (due to missing threads).
-



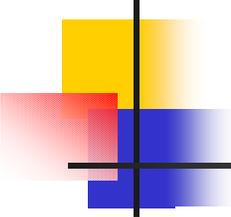
To Do List (Sorted by Priority)

- Implement threads.
- Clean the code - lot of work...sorry.
- Check OpenGL.
- Port X3d in the same way than TGX11.
- Extract the code needed by ROOT from gdk - big job.
- ...



Foresights

- First release ? good question...next question please.
- Seriously, I think if I could spend 100% of my time on it, in collaboration with ROOT team, I should be able to announce the first release in a period of 4-5 weeks, depending of the time to solve the threads problem.
But my day (and sometimes night) job takes me already more than 100% of my time, so you can understand that it is very difficult for me to give precise timetable. What I already know is that for the next six weeks, I will be alone at work, so don't expect too much from me during this period.



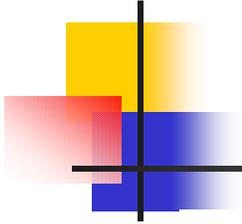
Annexes

- Needs.
 - If someone knows well win32, threads and/or gdk, his/her help is welcome.
- Questions - Infos.
 - Feel free to contact me at my professional or private e-mail addresses :
bertrand.bellenot@alusuisse.com
bellbert@scopus.ch.

More informations

Status and progress reports will be on the Root Win32 GUI web page :

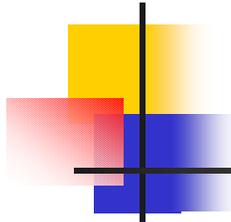
<http://root.cern.ch/root/win32progress/Win32GUI.html>



Thank's to ROOT team for
their great job.

Thank's to FNAL people for
this workshop.

And thank you everybody for
your patience.



Questions

- If you ask a stupid question, you may feel stupid; if you don't ask a stupid question, you remain stupid...