Graphics User Interface in ROOT

Ilka Antcheva

Overview

- ROOT GUI Classes
  - TVirtualX
  - Widgets
  - Layout Managers
  - Signals/Slots Communication
- Developing a GUI
  - GUI Code Generation
- Pad Editor
  - General Features
  - Focus on Users
- Next Steps

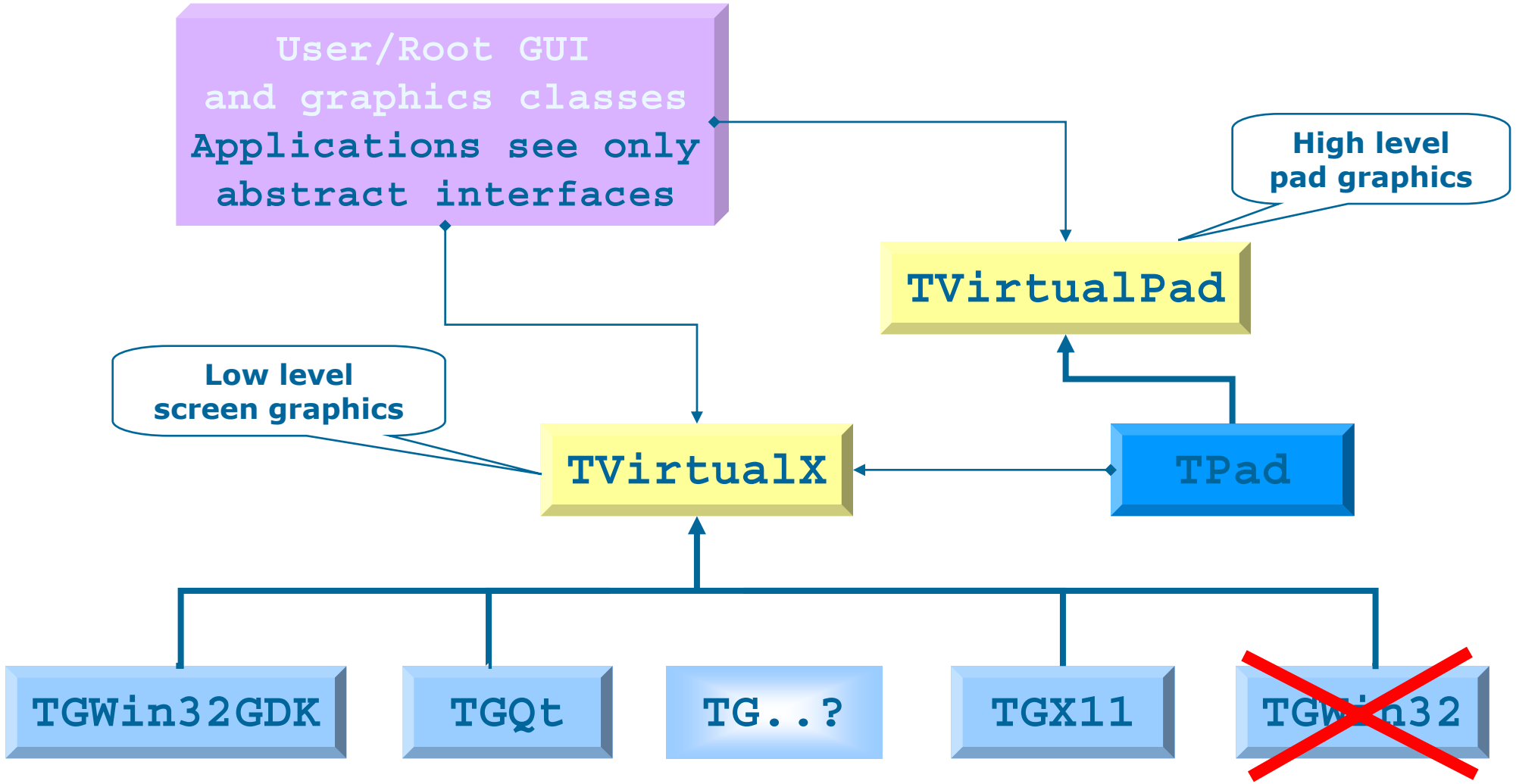
ROOT GUI Classes

- Provide standard components for a GUI environment

  Based on the XClass library from Hector Peraza

  - Containers - use of ROOT container classes for fast object look up
  - Controls / widgets
  - Layout managers
- Object-oriented, event-driven programming model
  - Can be extended via inheritance
  - Signals/slots communication
  - Conventional model
- Important classes
  - TGClient – sets up the graphics system
  - TGResourcePool – global resource manager

TVirtualX


User/Root GUI
and graphics classes
Applications see only
abstract interfaces
High level
pad graphics
TVirtualPad
Low level
screen graphics
TVirtualX
TPad
TGWin32GDK
TGQt
TG..?
TGX11
TGWin32

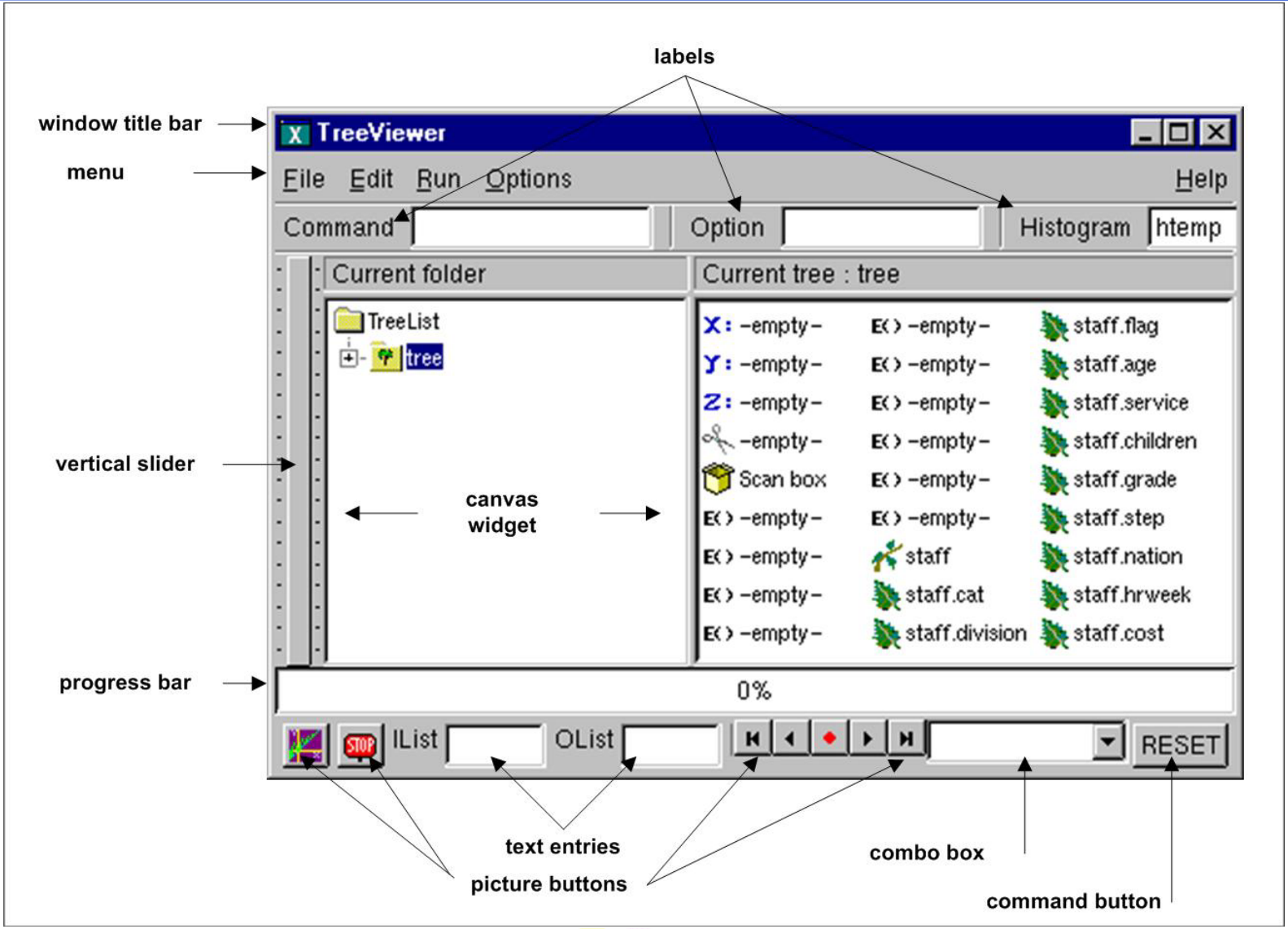
Widgets (1)

- Base classes
  - TGObject – window identifier; connection to the graphics system
  - TGWidget – important for event processing
  - TGWindow – creates a new window with common characteristics
  - TGFrame – base class for simple widgets
- GUI widgets
  - Buttons: text, picture, check, radio
  - Menus: menu bar, menu title, popup menu, tool bar
  - Containers: combo box, list box, tabs, shutter
  - Text widgets: label, text entry, number entry, text edit, tool tip
  - Color: color selector, dialog, palette
  - Whiteboard: canvas, list tree
  - Miscellaneous: slider, splitter, scroll bar, spin box, progress bar, 3Dlines

Widgets (2)

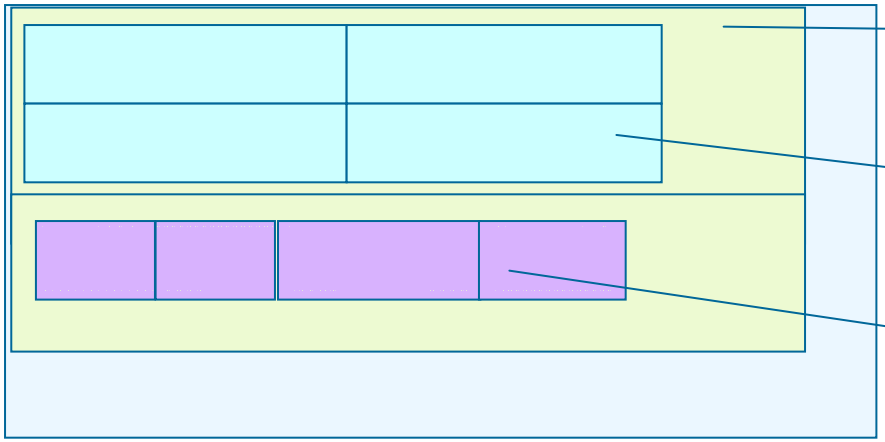

TGWindow
TGFrame
Simple widgets
Label, scroll bar, icon, menu objects, splitter, slider, text entry, etc.
TGCompositeFrame
TGCanvas
Whiteboard
Embedded canvas, list view
TGButton
Button widgets
Command and option buttons
Container widgets
Combo box, list box, tool bar, double slider shutter, etc.
TGGroupFrame
TGMainFrame
TGHorizontalFrame
TGVerticalFrame
TGTransientFrame
Dialog widgets
File dialog, help dialog, search dialog, print dialog message box, etc.
ROOT interface
Browser, control bar, canvas window with menus

Widgets (3)


labels
window title bar
menu
vertical slider
progress bar
TreeViewer
File Edit Run Options Help
Command
Option
Histogram
htemp
Current folder
Current tree : tree
TreeList
tree
canvas widget
staff.flag
staff.age
staff.service
staff.children
staff.grade
staff.step
staff.nation
staff.hrweek
staff.cost
staff
staff.cat
staff.division
Scan box
0%
IList
OList
RESET
text entries
picture buttons
combo box
command button

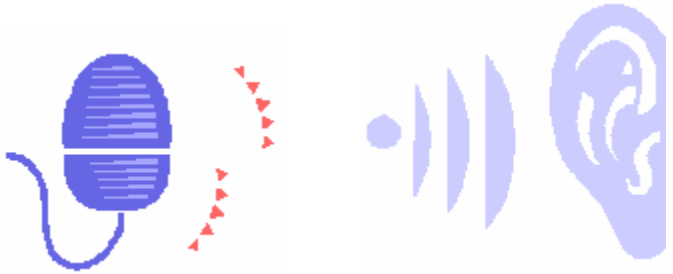
Layout Managers

- Laying out the GUI components
  - Control the arrangement of components within a container
  - Parent-children relationship between widgets
  - Different managers
  - Layout hints specify a widget position in the container



Signals/Slots Communication (1)

- Signals and slots pioneered by Trolltech Qt GUI toolkit
- Integrated into the ROOT core
  - TQObject, TQConnection, TQClass



- Facilitates the programming since it allows a total independence of the interacting classes
- Classes emit signals whenever they change a significant state that others might be interested in

Signals/Slots Communication (2)

- Widgets send out various *signals*
  - Event senders

```
virtual void Toggled(Bool_t on) { Emit("Toggled(Bool_t)", on); } // *SIGNAL*
```

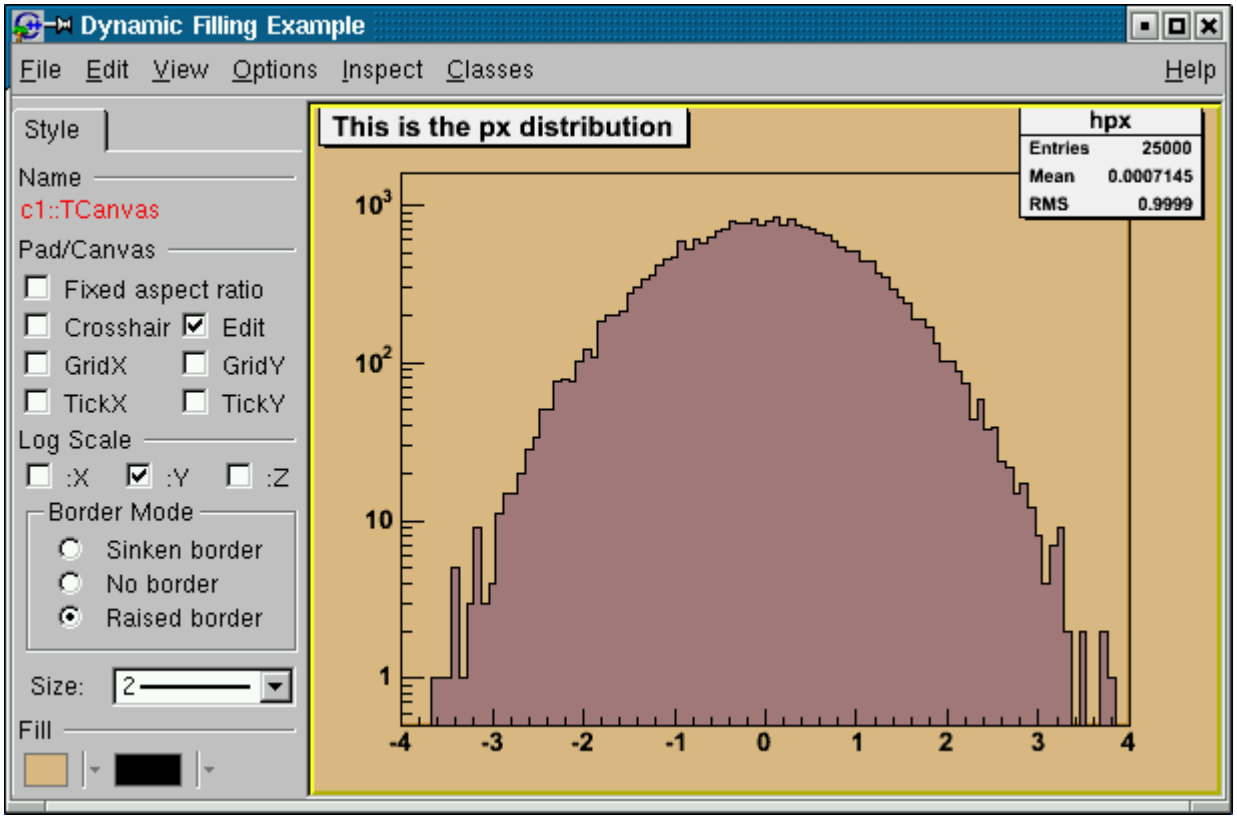
- Object methods can be used as *slots*
  - Event receivers

```
void TPadEditor::DoLogX(Bool_t on)
{  fPadPointer->SetLogx(on);   Update();  }
```

- Signals and slots can be connected together
  - Use the dictionary information and CINT interpreter to connect signals to slots

```
fLogX->Connect("Toggled(Bool_t)","TPadEditor",this,"DoLogX(Bool_t)");
```

Signals/Slots Communication (3)



*virtual void Toggled(Bool_t on) { Emit("Toggled(Bool_t)", on); } // *SIGNAL**

void TPadEditor::DoLogY(Bool_t on)
{ fPadPointer->SetLogy(on); Update(); }

fLogX->Connect("Toggled(Bool_t)","TPadEditor",this,"DoLogY(Bool_t)");

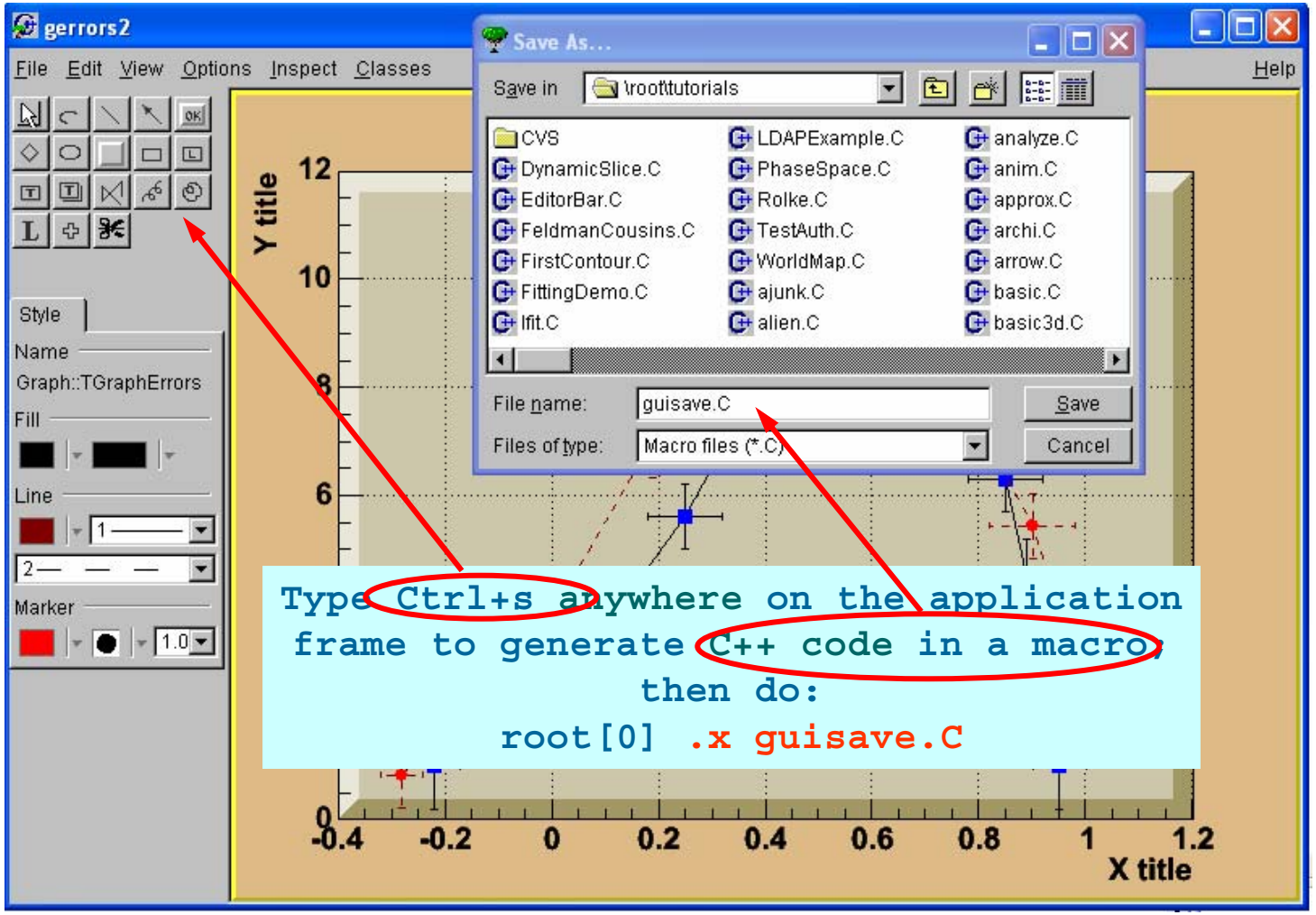
Developing a GUI

- Selecting widgets provided by ROOT GUI classes
  - Rich set for standard application functionality
  - Intuitive naming conventions
  - *SavePrimitive()* methods
- Laying out the GUI components
  - Parent-children relationship between widgets
  - Different layout managers
- Programming the components to perform actions
  - Signals/slots mechanism
  - Conventional model of event processing
- Running the GUI

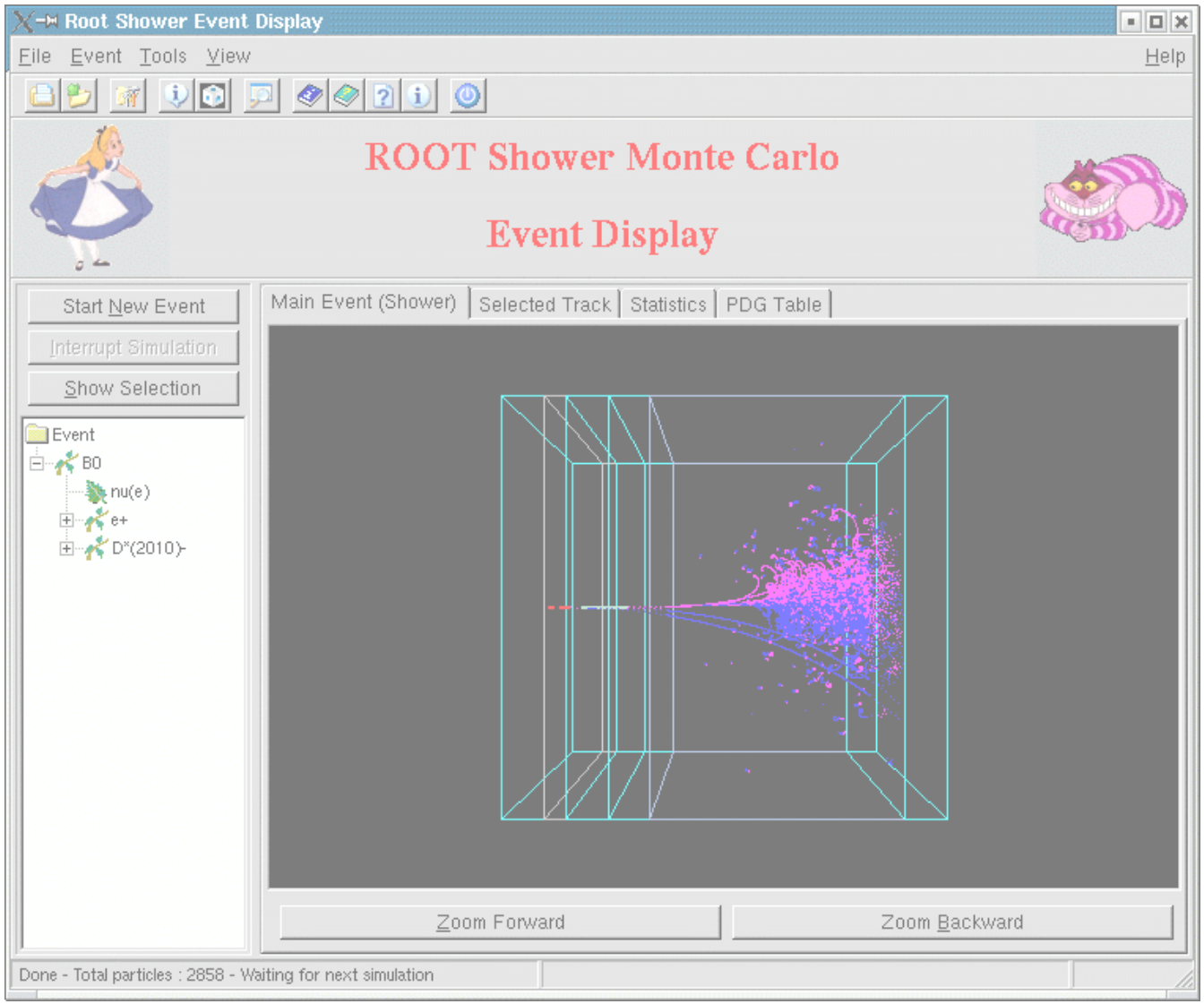
GUI Code Generation (1)

- Methods *SavePrimitive()* generate automatically a macro after *Ctrl+s*
- Executing this macro brings back the GUI widgets with their attributes and keeps the layout without the action handling based on the process event or signals/slots mechanism
- Saving a running application: *TGMainFrame::SaveSource("macro.C","")*
- Saving a dialog: *TGTransientFrame::SaveSource("dialogMacro.C","")*

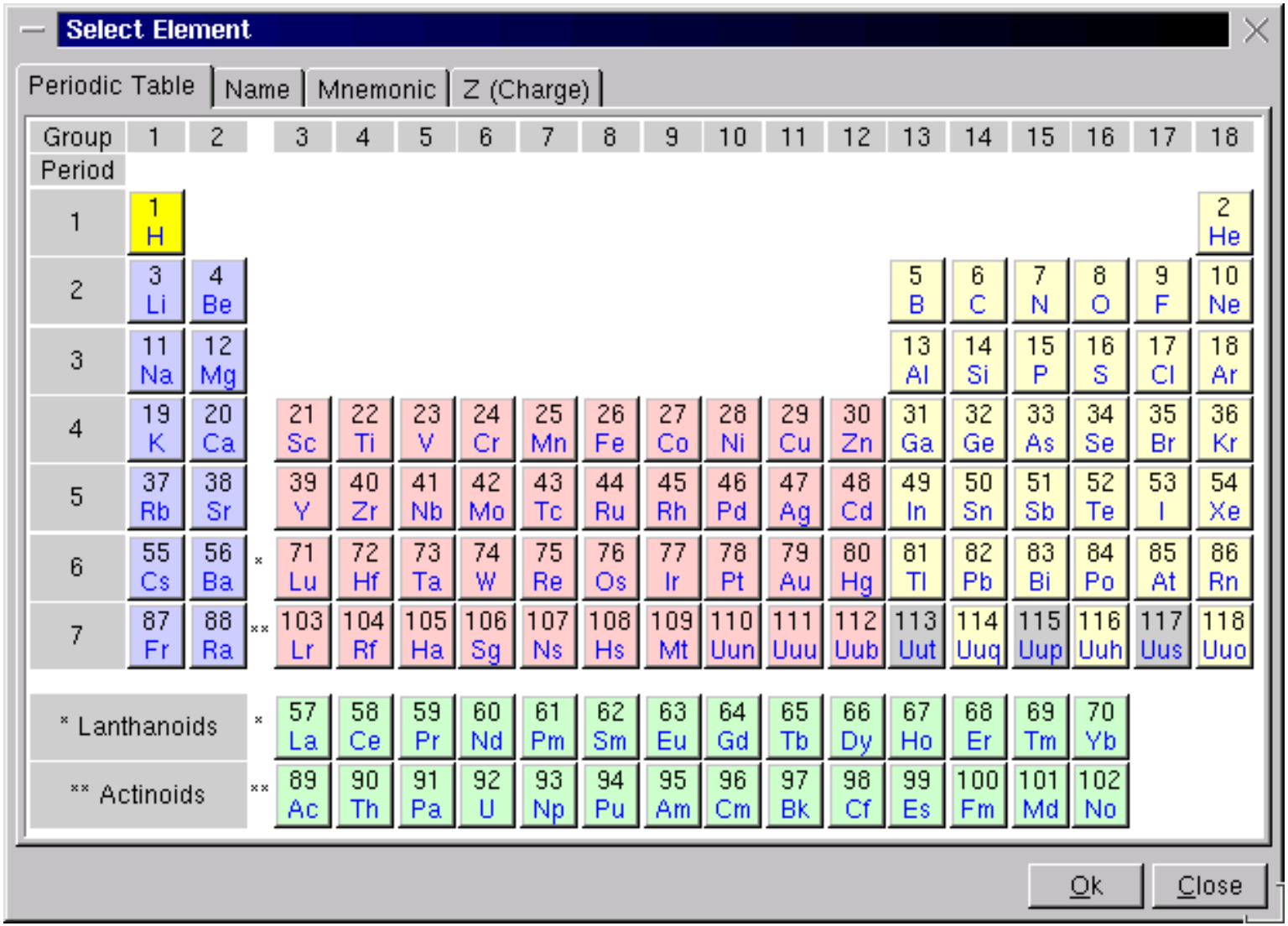
GUI Code Generation (2)



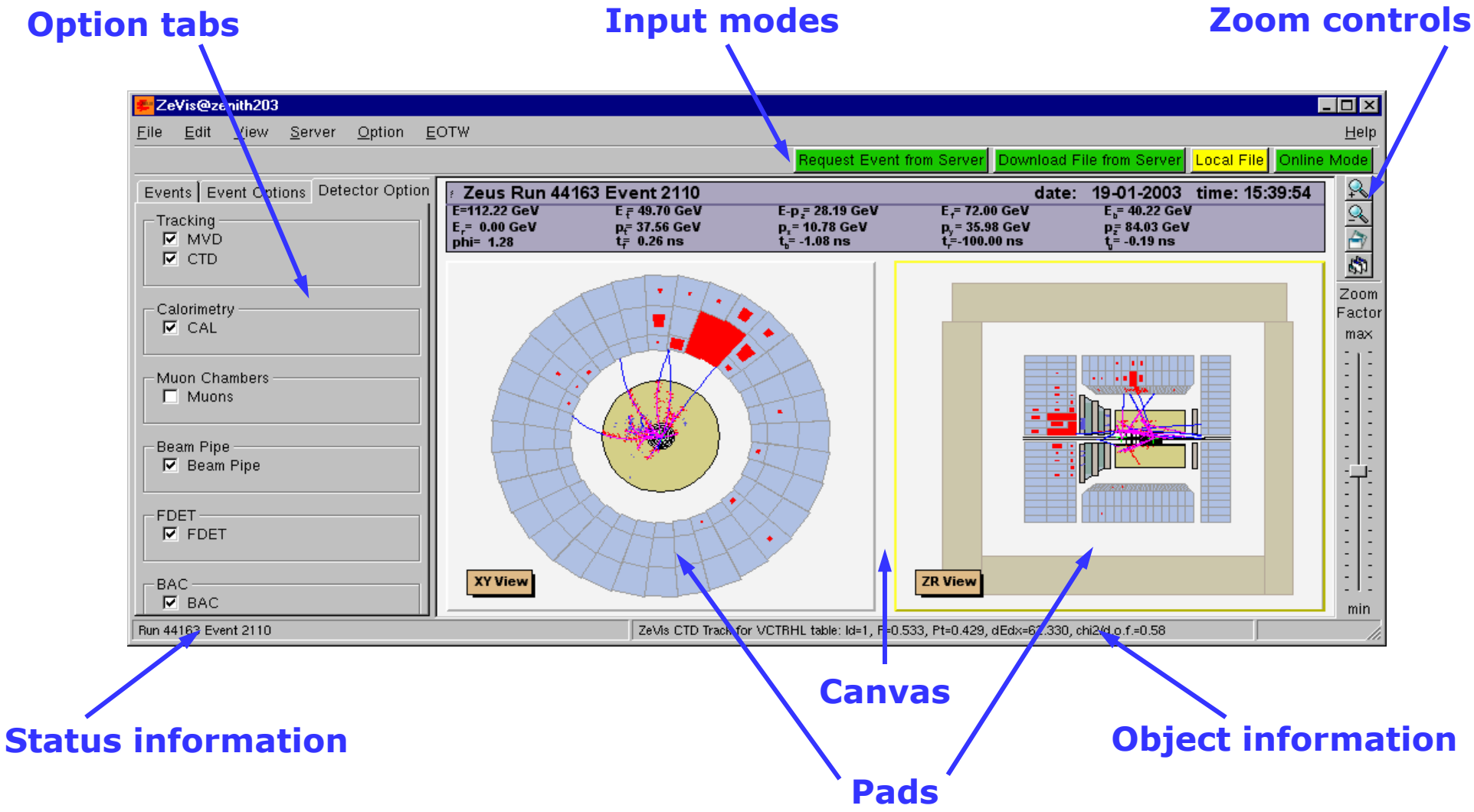
Examples (1)


Root Shower Event Display
File Event Tools View Help
ROOT Shower Monte Carlo
Event Display
Start New Event
Interrupt Simulation
Show Selection
Event
B0
nu(e)
e+
D*(2010)-
Main Event (Shower)
Selected Track
Statistics
PDG Table
Zoom Forward
Zoom Backward
Done - Total particles : 2858 - Waiting for next simulation

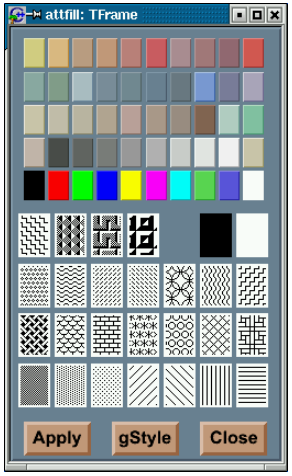
Examples (2)

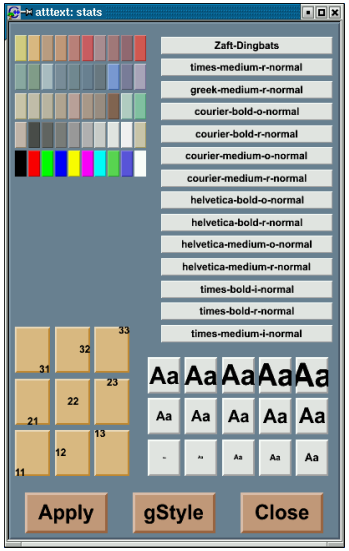

Select Element
Periodic Table
Name
Mnemonic
Z (Charge)
Group 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Period
1 1 H 2 He
2 3 Li 4 Be 5 B 6 C 7 N 8 O 9 F 10 Ne
3 11 Na 12 Mg 13 Al 14 Si 15 P 16 S 17 Cl 18 Ar
4 19 K 20 Ca 21 Sc 22 Ti 23 V 24 Cr 25 Mn 26 Fe 27 Co 28 Ni 29 Cu 30 Zn 31 Ga 32 Ge 33 As 34 Se 35 Br 36 Kr
5 37 Rb 38 Sr 39 Y 40 Zr 41 Nb 42 Mo 43 Tc 44 Ru 45 Rh 46 Pd 47 Ag 48 Cd 49 In 50 Sn 51 Sb 52 Te 53 I 54 Xe
6 55 Cs 56 Ba * 71 Lu 72 Hf 73 Ta 74 W 75 Re 76 Os 77 Ir 78 Pt 79 Au 80 Hg 81 Tl 82 Pb 83 Bi 84 Po 85 At 86 Rn
7 87 Fr 88 Ra ** 103 Lr 104 Rf 105 Ha 106 Sg 107 Ns 108 Hs 109 Mt 110 Uun 111 Uuu 112 Uub 113 Uut 114 Uuq 115 Uup 116 Uuh 117 Uus 118 Uuo
* Lanthanoids * 57 La 58 Ce 59 Pr 60 Nd 61 Pm 62 Sm 63 Eu 64 Gd 65 Tb 66 Dy 67 Ho 68 Er 69 Tm 70 Yb
** Actinoids ** 89 Ac 90 Th 91 Pa 92 U 93 Np 94 Pu 95 Am 96 Cm 97 Bk 98 Cf 99 Es 100 Fm 101 Md 102 No
Ok
Close

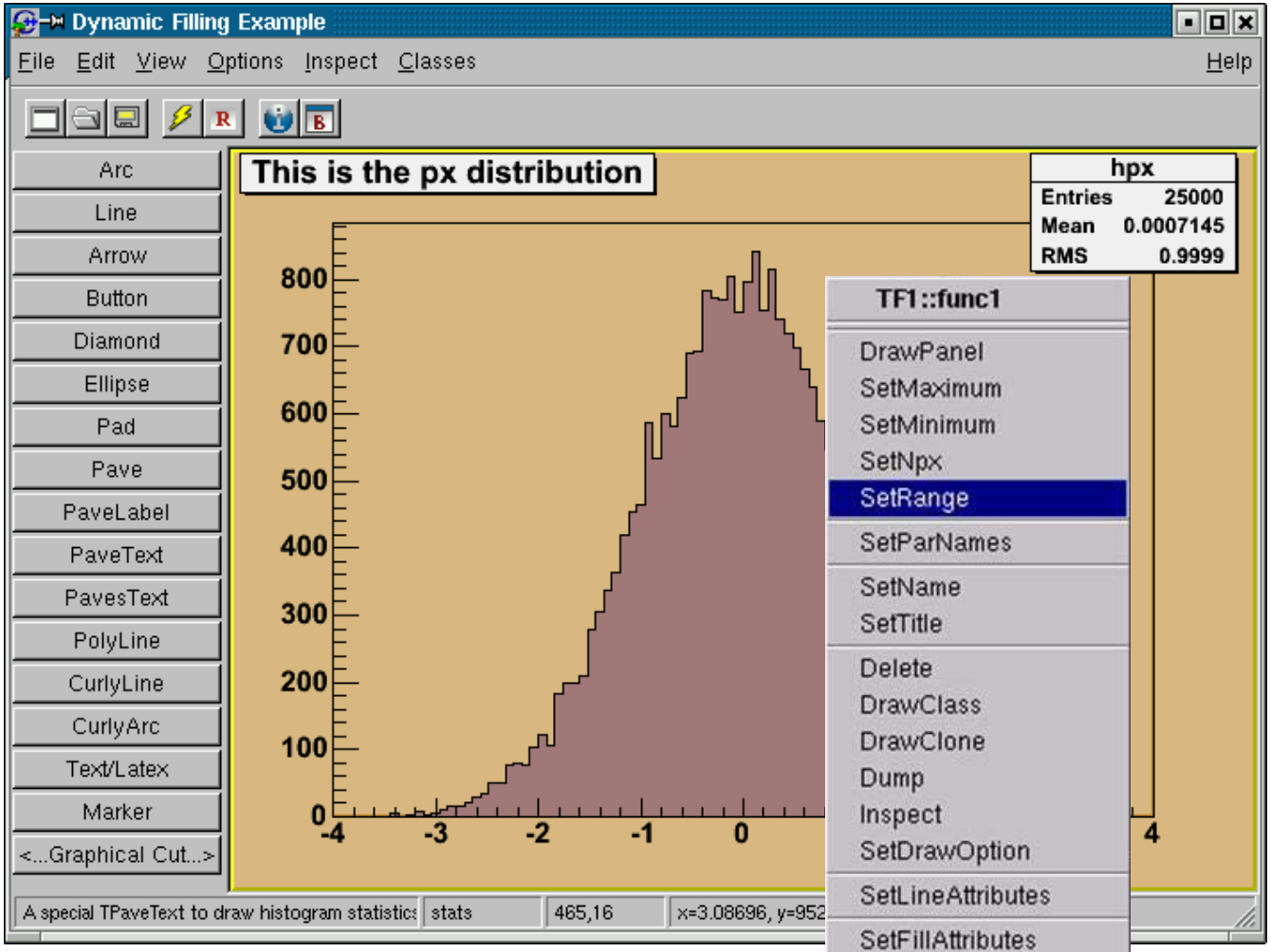
Examples (3)

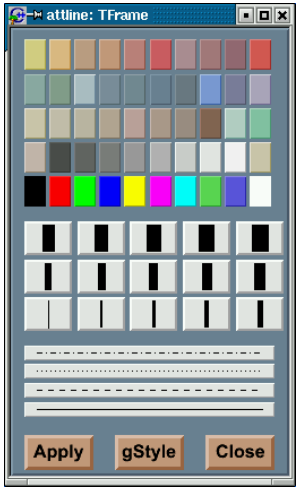

Option tabs
Input modes
Zoom controls
ZeVis@zenith203
File Edit View Server Option EOTW Help
Request Event from Server
Download File from Server
Local File
Online Mode
Events
Event Options
Detector Option
Tracking
MVD
CTD
Calorimetry
CAL
Muon Chambers
Muons
Beam Pipe
Beam Pipe
FDET
FDET
BAC
BAC
Zeus Run 44163 Event 2110
date: 19-01-2003 time: 15:39:54
E=112.22 GeV
E_f= 49.70 GeV
E-p_z= 28.19 GeV
E_r= 72.00 GeV
E_b= 40.22 GeV
E_r= 0.00 GeV
p_t= 37.56 GeV
p_x= 10.78 GeV
p_y= 35.98 GeV
p_z= 84.03 GeV
phi= 1.28
t_f= 0.26 ns
t_b= -1.08 ns
t_r=-100.00 ns
t_v= -0.19 ns
XY View
ZR View
Zoom Factor
max
min
Run 44163 Event 2110
ZeVis CTD Track for VCTRHL table: Id=1, Pt=0.429, dEdx=6
Canvas
Status information
Pads
Object information

Old Pad Editor





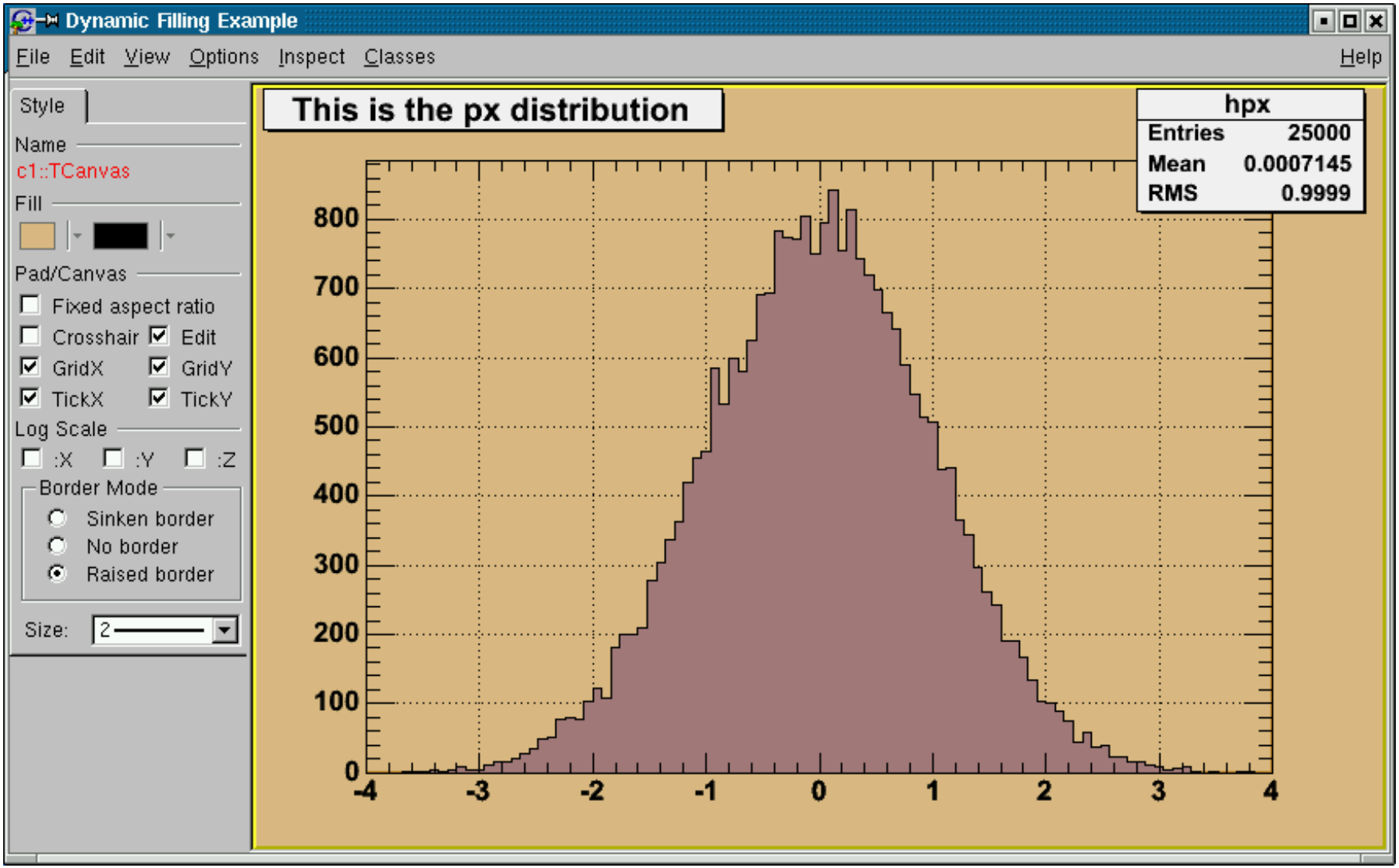




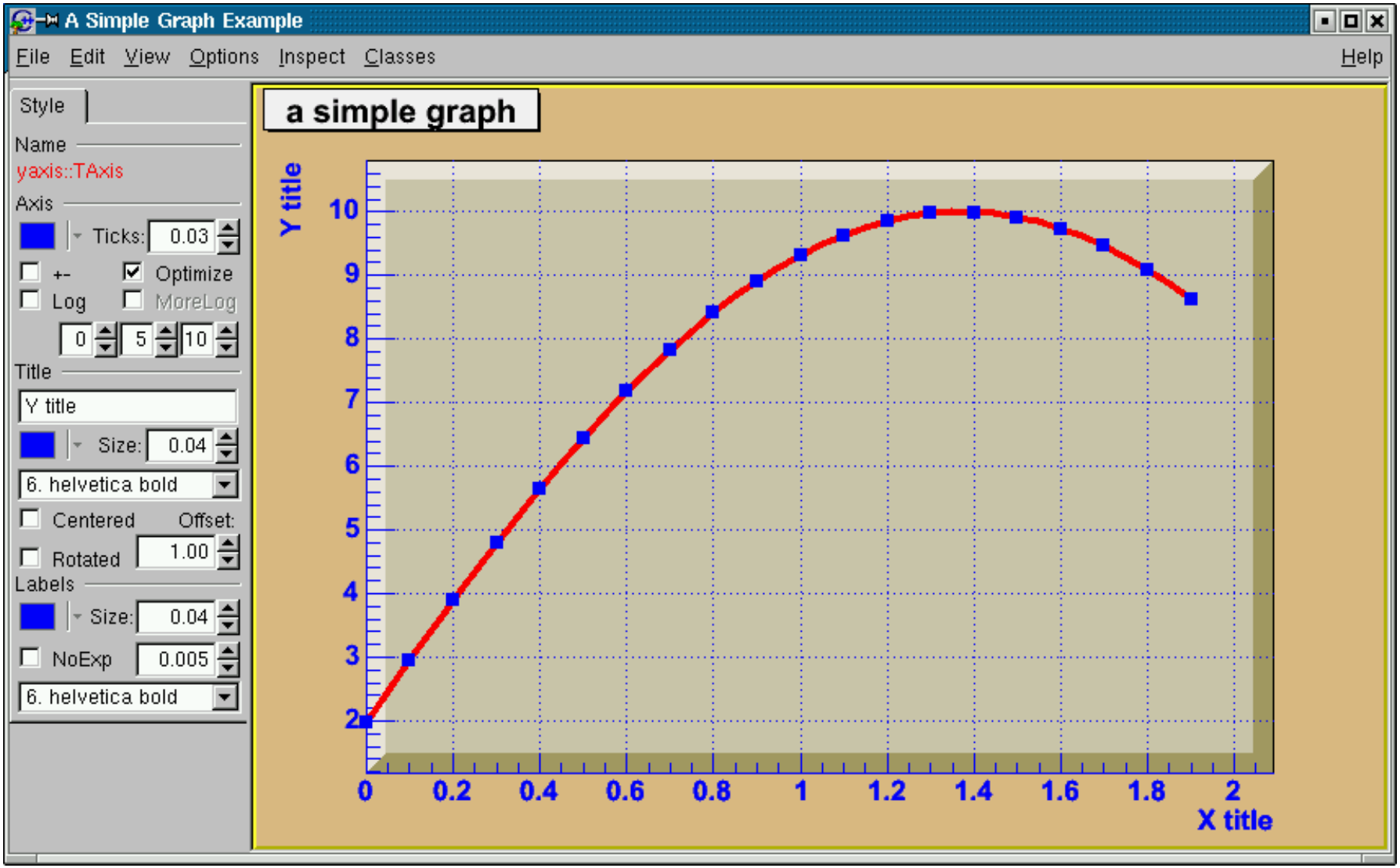
New Pad Editor (1)

- Visual composition draws attention clearly and quickly
- Consistency – related actions work the same way
- Direct manipulation and immediate feedback
- Forgiveness – prevent/inform about dangerous steps
- Persistence – keep the same environment look
- Control – one possible thing to do (*the right one*)
- Efficiency – minimize users' work
  - *Recognition, not recall*
  - *Three mouse clicks*
- Predictability - reduce short-term memory load
  - *Miller's law of 7*

New Pad Editor (2)



New Pad Editor (3)



Focus on Users

- Novices *(for a short time)*
  - Theoretical understanding, no practical experience with ROOT
  - Impatient with learning concepts; patient with performing tasks
- Advanced beginners *(many people remain at this level)*
  - Focus on a few tasks and learn more on a need-to-do basis
  - Perform several given tasks well
- Competent performers *(fewer then previous class)*
  - Know and perform complex tasks that require coordinated actions
  - Interested in solving problems and tracking down errors
- Experts *(identified by others)*
  - Ability to find solution in complex functionality
  - Interested in theories behind the design
  - Interested in interacting with other expert systems

Next Steps

- GUI for editing objects in ROOT
  - Histogram editor & fitter by Carsten Hof
  - Graphics primitives
- Improve the design according to the users' feedback
- Modify and iterate as much as necessary
- Integrate all system components
  - Software
  - Documentation
  - Help functions
  - Training tools