Lecture 3

Finding your way in ROOT memory:
Names, Lists, Directories, Browsers and Files

Exercises

① Use the browser to find out *which* standard presentation styles are available in ROOT. Hint: there are 5

Exercises

① Use the browser to find out *which* standard presentation styles are available in ROOT. Hint: there are 5

Solution:

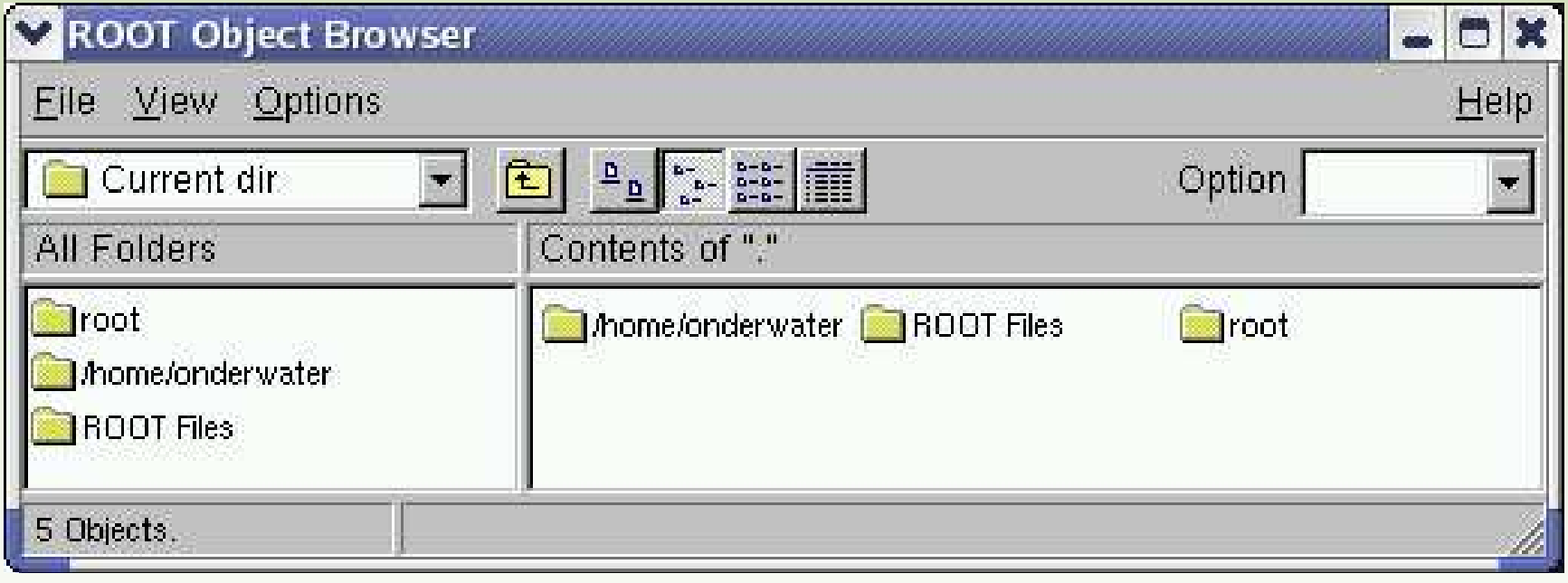
```
KVIQ75:tmp:906>root -l
root [0] new TBrowser
(class TBrowser*)0x8c758a8
```

Exercises

① Use the browser to find out *which* standard presentation styles are available in ROOT. Hint: there are 5

Solution:

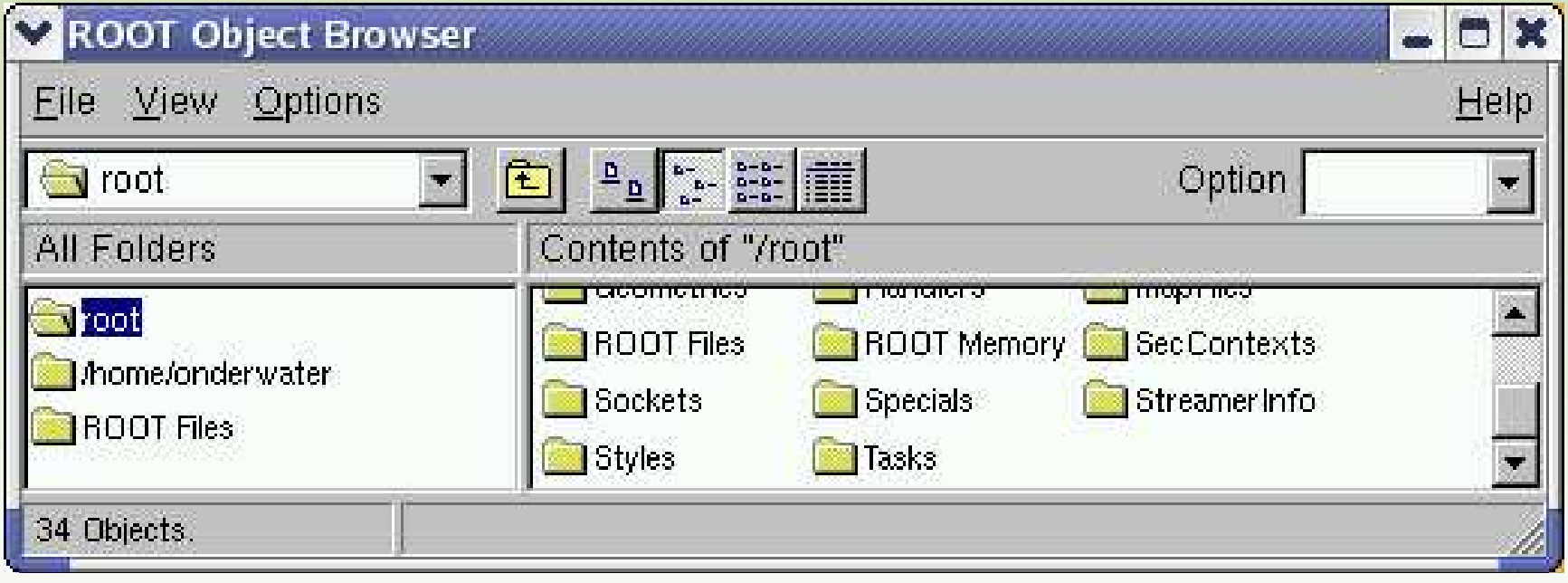
KVIQ75:tmp:906>root -l
root [0] new TBrowser
(class TBrowser*)0x8c758a8



Exercises

① Use the browser to find out *which* standard presentation styles are available in ROOT. Hint: there are 5

Solution:
KVIQ75:tmp:906>root -l
root [0] new TBrowser
(class TBrowser*)0x8c758a8

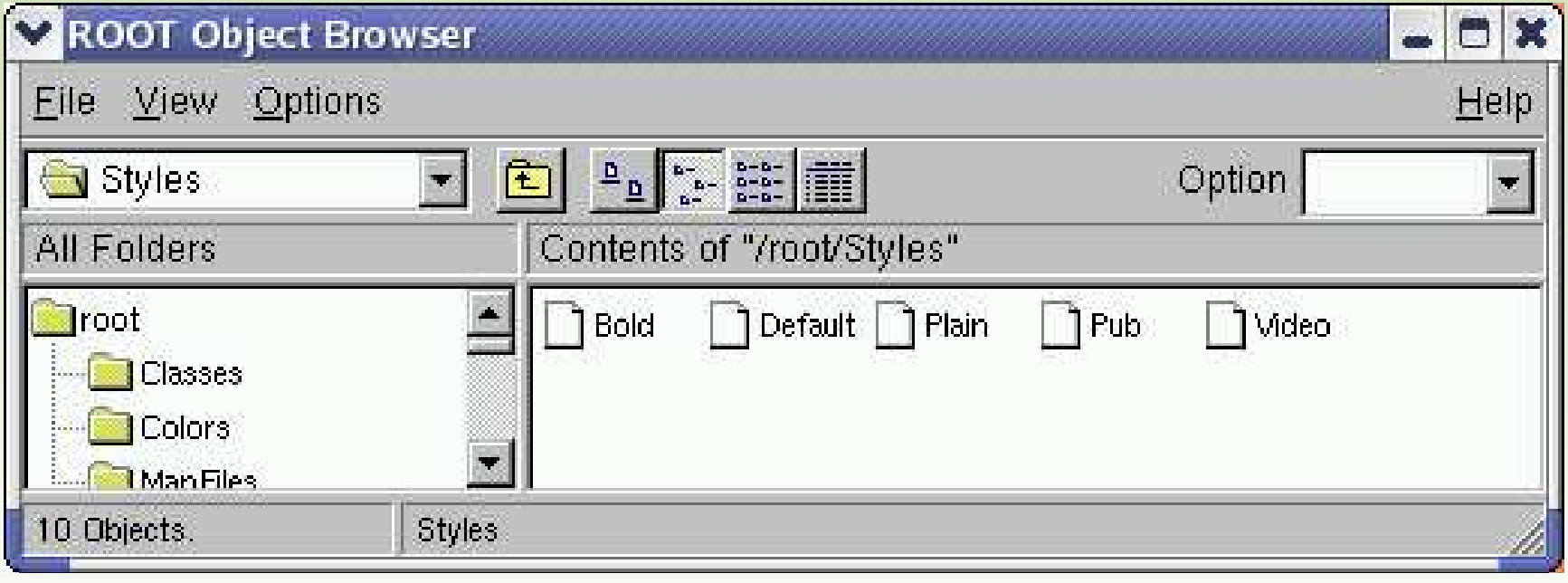


Exercises

① Use the browser to find out *which* standard presentation styles are available in ROOT. Hint: there are 5

Solution:

KVIQ75:tmp:906>root -l
root [0] new TBrowser
(class TBrowser*)0x8c758a8



Exercises

② Read http://root.cern.ch/root/htmldoc/TFile.html and the ROOT tutorial #6 on the web. Create a file with a histogram in it. Make sure you close the file. Start ROOT again and open the file you just created with the browser. See if the histogram is indeed there ....

Exercises

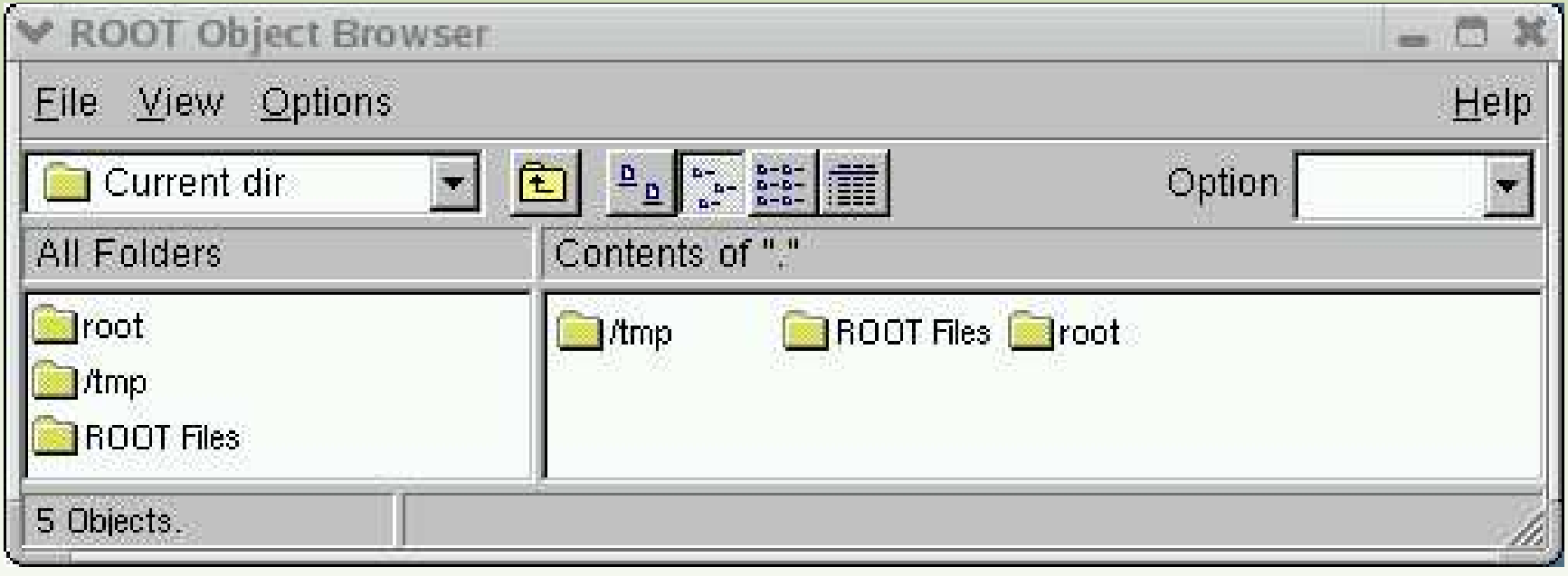
② Read http://root.cern.ch/root/htmldoc/TFile.html and the ROOT tutorial #6 on the web. Create a file with a histogram in it. Make sure you close the file. Start ROOT again and open the file you just created with the browser. See if the histogram is indeed there ....

Solution:

```
KVIQ75:tmp:910>root -l
root [0] TFile* file = new TFile("test.root","create")
root [1] TH1D* myHistoPtr = new TH1D("histName","histTitle",100,0,1)
root [2] file->Write()
(Int_t)249
root [3] file->Close()
root [4] .q
KVIQ75:tmp:912>root -l
root [0] new TBrowser
(class TBrowser*)0x8c758a8
```

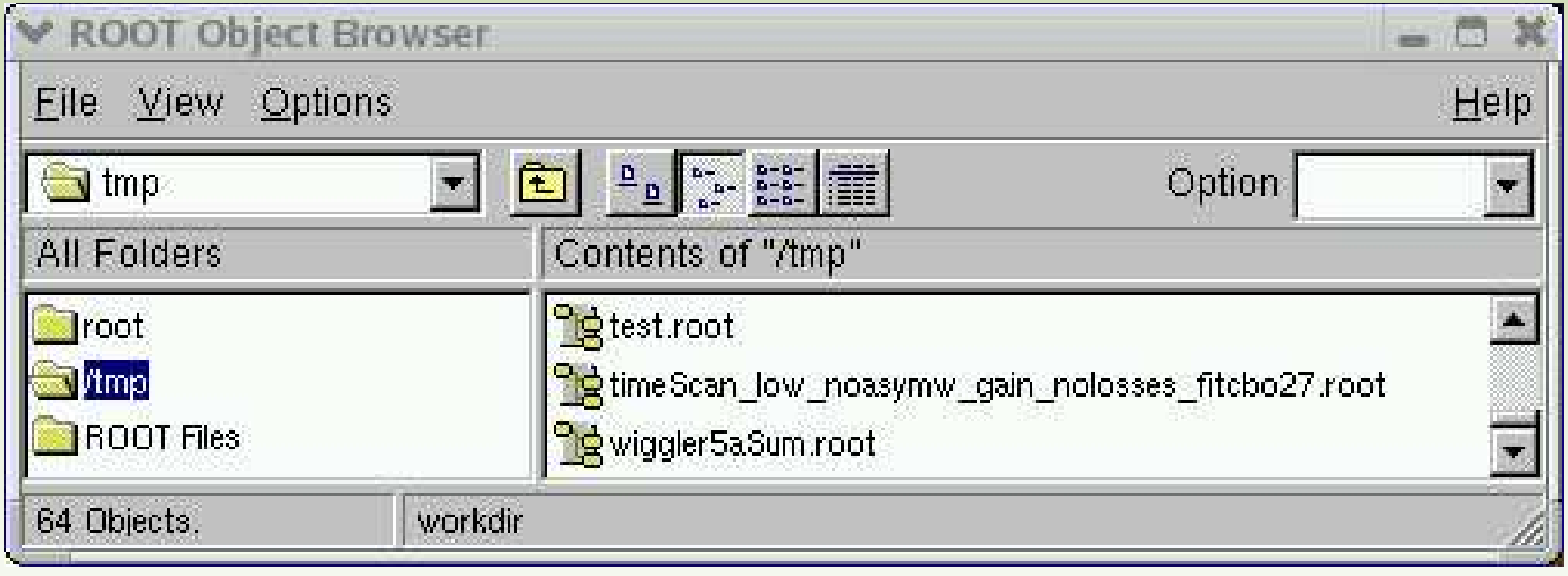
Exercises

② Read http://root.cern.ch/root/htmldoc/TFile.html and the ROOT tutorial #6 on the web. Create a file with a histogram in it. Make sure you close the file. Start ROOT again and open the file you just created with the browser. See if the histogram is indeed there ....



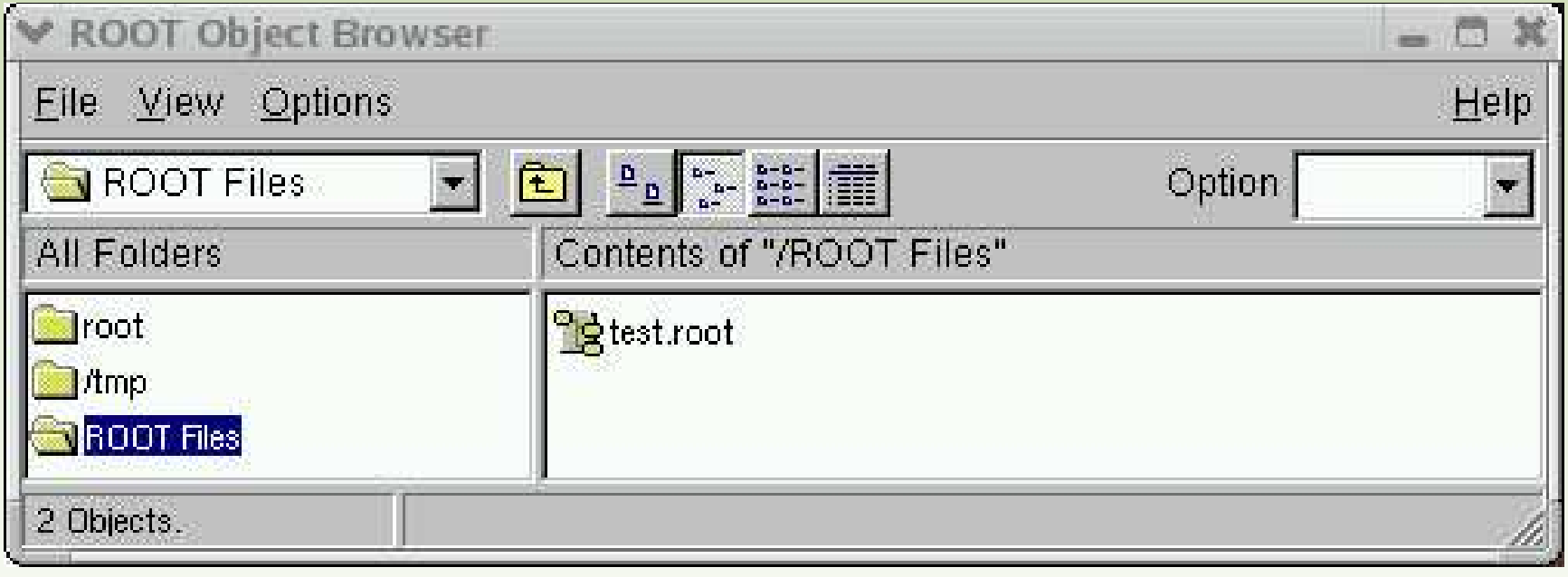
Exercises

② Read http://root.cern.ch/root/htmldoc/TFile.html and the ROOT tutorial #6 on the web. Create a file with a histogram in it. Make sure you close the file. Start ROOT again and open the file you just created with the browser. See if the histogram is indeed there ....



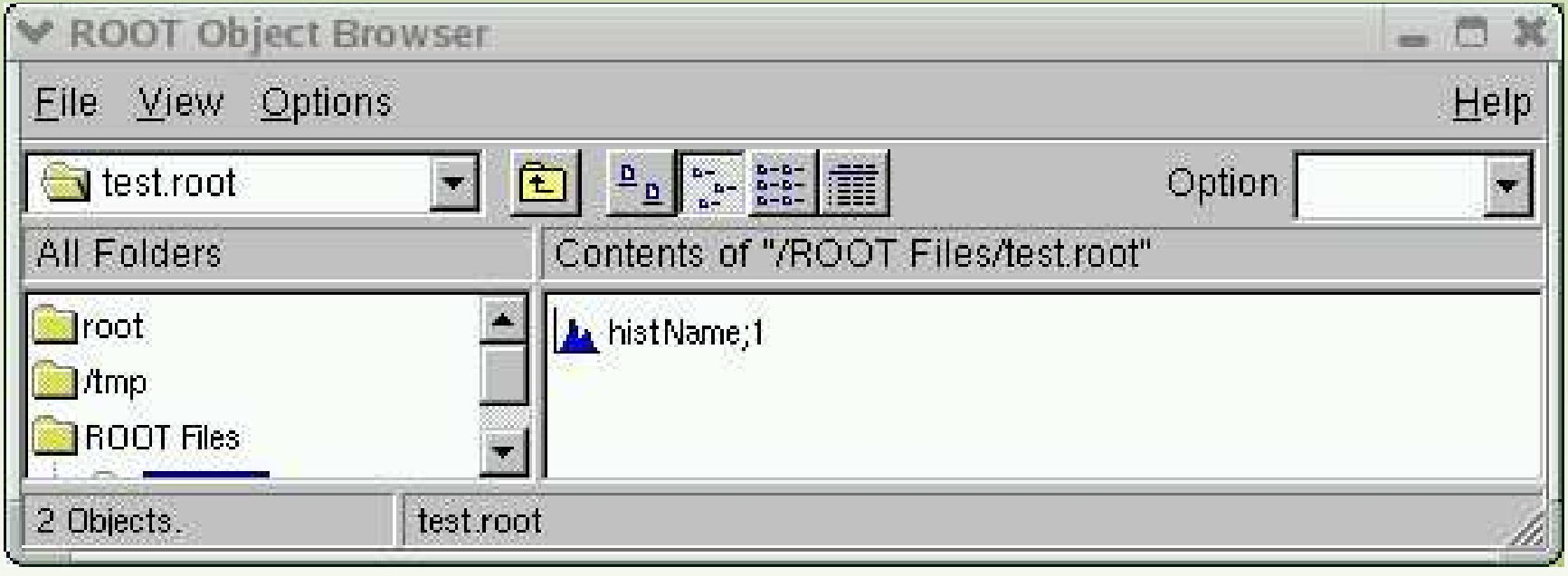
Exercises

② Read http://root.cern.ch/root/htmldoc/TFile.html and the ROOT tutorial #6 on the web. Create a file with a histogram in it. Make sure you close the file. Start ROOT again and open the file you just created with the browser. See if the histogram is indeed there ....



Exercises

② Read http://root.cern.ch/root/htmldoc/TFile.html and the ROOT tutorial #6 on the web. Create a file with a histogram in it. Make sure you close the file. Start ROOT again and open the file you just created with the browser. See if the histogram is indeed there ....



Exercises

③ Open the file you created in (2) in *update* mode and change the title of the histogram. Describe what you did.

Exercises

③ Open the file you created in (2) in *update* mode and change the title of the histogram. Describe what you did.

Solution:

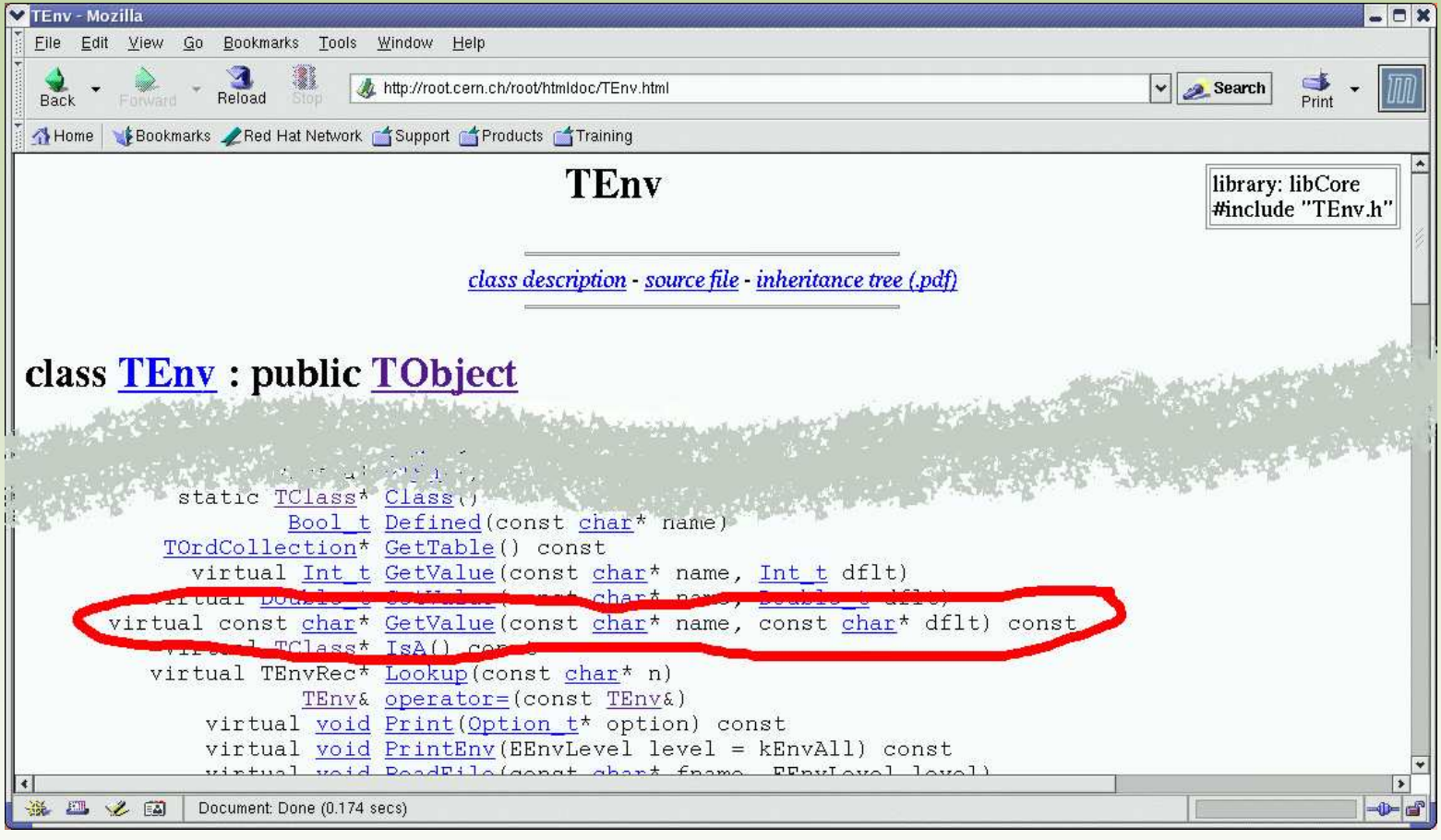
```
KVIQ75:tmp:912>root -l
root [0] TFile* f = new TFile("test.root","update")
root [1] TH1D* hist = (TH1D*)f->Get("histName")
root [2] hist->SetName("newTitle")
root [3] f->Write()
(Int_t)250
root [4] f->Close() KVIQ75:tmp:913>root -l
root [0] new TBrowser
(class TBrowser*)0x8c758a8
```

Exercises

④ From gEnv (an instance of the TEnv class), get the name of the default fitter in ROOT: "Root.Fitter". Hint: for *dflt* use ""

Exercises

④ From gEnv (an instance of the TEnv class), get the name of the default fitter in ROOT: "Root.Fitter". Hint: for *dflt* use ""



Exercises

④ From gEnv (an instance of the TEnv class), get the name of the default fitter in ROOT: "Root.Fitter". Hint: for *dflt* use ""

Solution:

```
KVIQ75:tmp:1186>root
root [0] gEnv->GetValue("Root.Fitter","")
(const char* 0x87ab17c)"Minuit"
```

Lecture 4

Graphs and Histograms

Graphs

A graph is a graphics object made of two arrays X and Y, holding the x, y coordinates of n points. There are several graph classes, *e.g.*: **TGraph**, **TGraphErrors** and **TGraphAsymmErrors**. There are also 2D versions: **TGraph2D** and **TGraph2DErrors**, with x, y and z coordinates

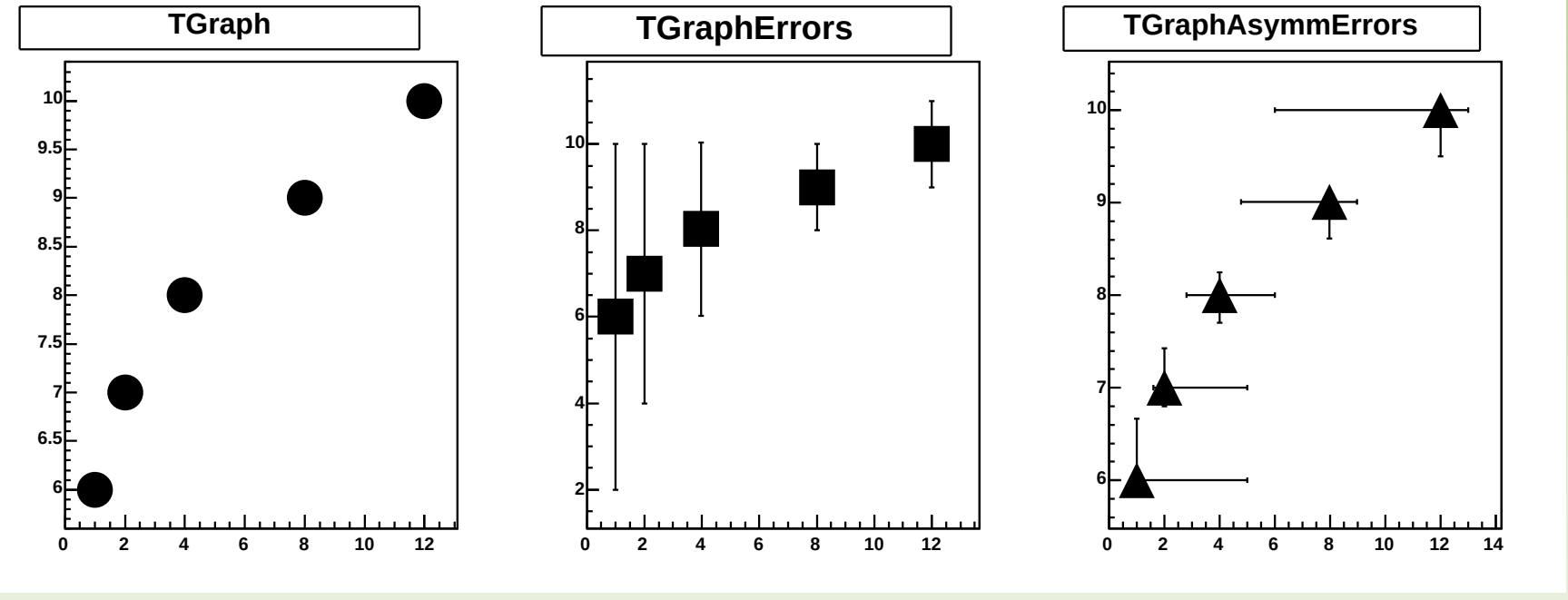
Constructors:

```
TGraph()
TGraph(Int_t n)
TGraph(Int_t n, const Int_t* x, const Int_t* y)
TGraph(Int_t n, const Float_t* x, const Float_t* y)
TGraph(Int_t n, const Double_t* x, const Double_t* y)
TGraph(const TGraph& gr)
TGraph(const TVector& vx, const TVector& vy)
TGraph(const TVectorD& vx, const TVectorD& vy)
TGraph(const TH1* h)
TGraph(const TF1* f, Option_t* option)
TGraph(const char* filename, const char* format = "%lg %lg", Option_t* option)
```

Example:

```
root [0] Float_t x[5] = {1,2,4,8,12};
root [1] Float_t y[5] = {6,7,8,9,10};
root [2] TGraph *gr1 = new TGraph (5, x, y);
```

Examples of Graphs

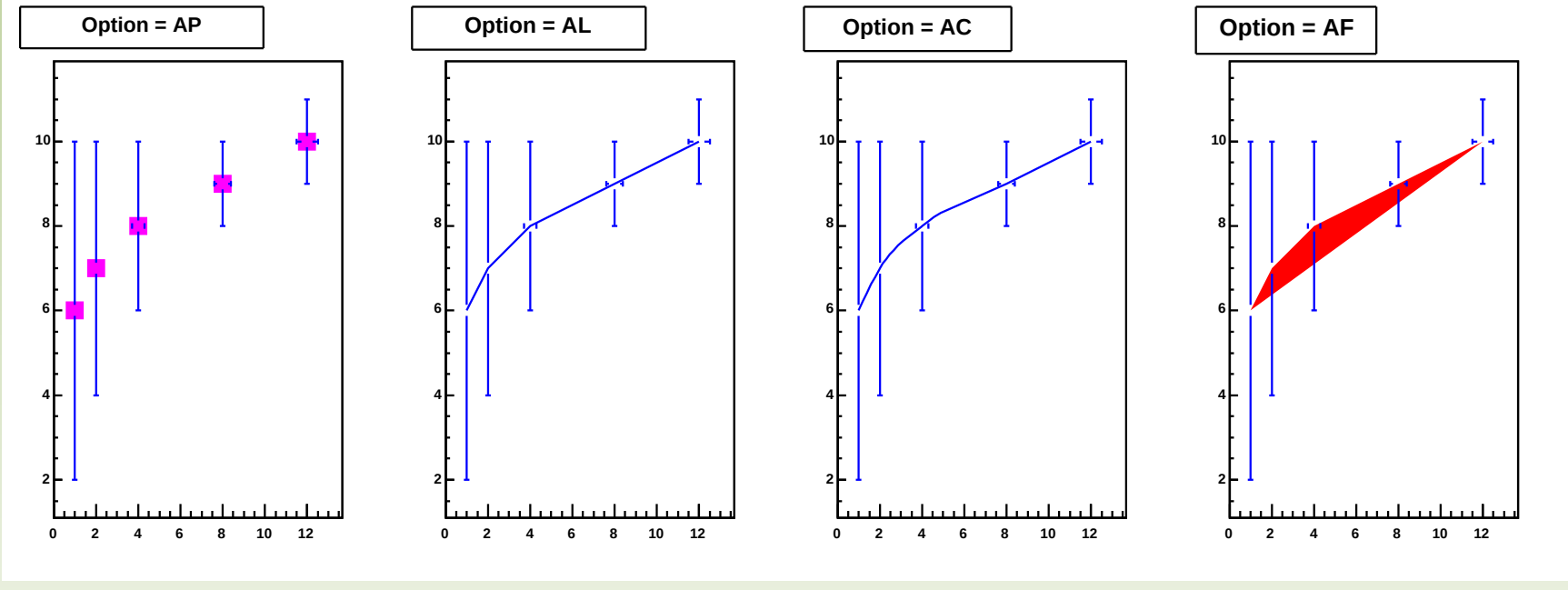


Drawing Options for Graphs

Graph Draw Options

- "L" A line between every points is drawn
- "F" A fill area is drawn
- "A" Axis are drawn around the graph (needed for stand-alone graph!!!)
- "C" A smooth curve is drawn
- "*" A star is plotted at each point
- "P" The current marker of the graph is plotted at each point
- "B" A bar chart is drawn at each point
- "[]" Only the end vertical/horizontal lines of the error bars are drawn. This option only applies to the TGraphAsymmErrors.

Examples of Graphs Drawing Options



Things You Can Do With Graphs

What?	How?
Print	graph->Print()
Set marker properties	graph->SetMarkerStyle(20)
	graph->SetMarkerColor(kRed)
	graph->SetMarkerSize(2)
Set line properties	graph->SetLineWidth(2)
	graph->SetLineStyle(2)
Set main title	graph->SetTitle("main-title")
Set axis titles	graph->GetXaxis()->SetTitle("axis-title")
	graph->GetYaxis()->SetTitle("axis-title")
Interpolate	graph->Eval(x)
Fit	graph->FitPanel()
	graph->Fit("function-name")
Calculate correlation	graph->GetCorrelationFactor()
Calculate covariance	graph->GetCovariance()
....	....

A lot more examples are in the user guide: `ftp://root.cern.ch/root/doc/chapter4.pdf`, the tutorials #25–31.

Histograms

There are several histograms classes available in ROOT, which contain data in the form of a number of (weighted) counts N for a collection of consecutive bins in x (1-D), (x, y) (2-D) or (x, y, z) (3-D). The corresponding classes are

THNS where $N = 1, 2, 3$ for 1-D, 2-D and 3-D and $S = "C, S, I, F, D"$

for 1 (Char_t), 2 (Short_t), 4 (Int_t), 4 (Float_t) or 8 (Double_t) bytes of storage volume per bin.

All histogram classes inherit from **TH1**.

Histogram Constructor

Contructors of **TH1D**:

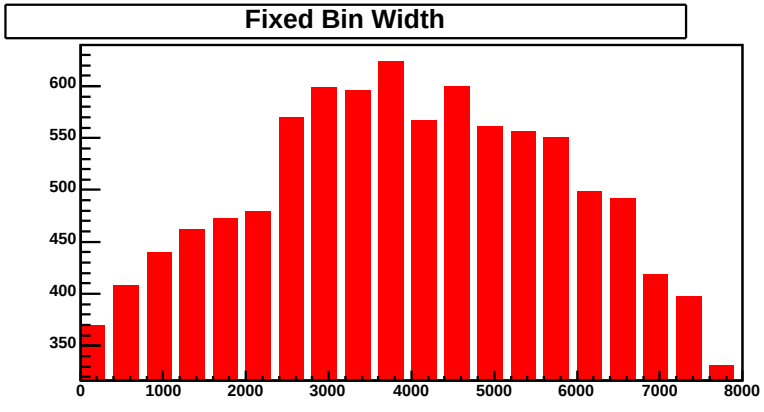
```
TH1D()
TH1D(const char* name, const char* title, Int_t nbinsx, Axis_t xlow, Axis_t xup)
TH1D(const char* name, const char* title, Int_t nbinsx, const Float_t* xbins)
TH1D(const char* name, const char* title, Int_t nbinsx, const Double_t* xbins)
TH1D(const TVectorD& v)
TH1D(const TH1D& h1d)
```

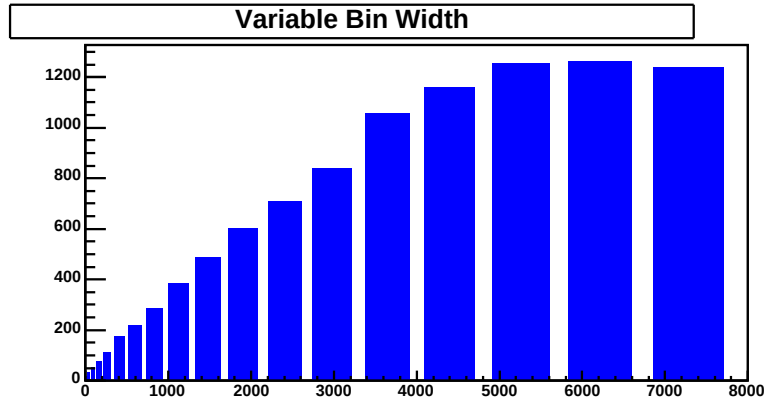
Example for fixed-bin width:

```
root [0] TH1D* hPtr = new TH1D("histoName","Fixed Bin Width",10,-1,1)
```

Example for variable-bin width:

```
root [0] Double_t bins[5] = {1,2,4,8,16}
root [1] TH1D* hPtr = new TH1D("histoName","Variable Bin Width",4,bins)
```



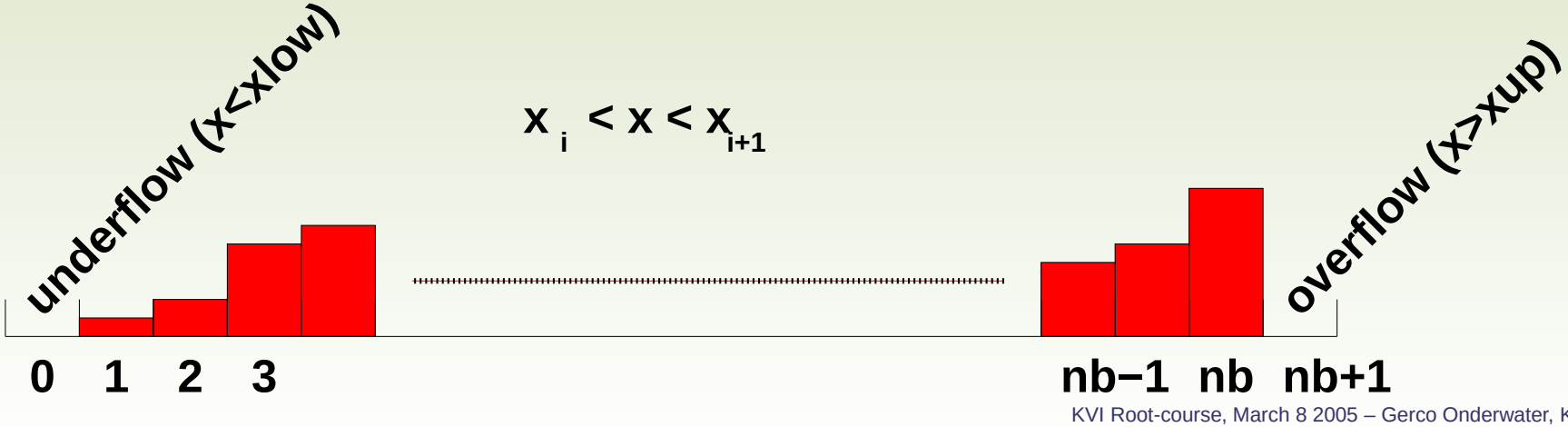


Conventions for Axes

You can look at the boundaries of the axes. Histogram axes are implemented via the **TAxis** class:

```
root [10] h2->GetXaxis()->GetBinLowEdge(1)
(const Axis_t)0.00000000000000000e+00
root [11] h2->GetXaxis()->GetBinUpEdge(1)
(const Axis_t)1.00000000000000000e+00
root [12] h2->GetXaxis()->GetBinCenter(1)
(const Axis_t)5.00000000000000000e-01
root [13] h2->GetXaxis()->GetBinWidth(1)
(const Axis_t)1.00000000000000000e+00
```

Bin Numbering Convention



Filling a Histogram

There are many ways to fill a histogram. A selection:

Method	**What happens?**
AddBinContent(Int_t bin)	increment content of ‘bin’ by 1
AddBinContent(Int_t bin, Stat_t w)	increment content of ‘bin’ by ‘w’
Eval(TF1* f1, Option_t* option)	by evaluation function ‘f1’ at bin centers
Fill(Axis_t x)	increment bin in which ‘x’ falls by 1
Fill(Axis_t x, Stat_t w)	increment bin in which ‘x’ falls by ‘w’
FillRandom(const char* fname, Int_t ntimes = 5000)	‘ntimes’ random events distributed as *fname*
Reset(Option_t* option)	set all contents to zero
SetBinContent(Int_t bin, Stat_t content)	set content of ‘bin’ to ‘content’

Inspecting Histogram Content

The simplest way to look at the histogram content is by drawing it
histo->Draw()

However, there are many things you can do from the command line:

Method	**What happens?**
histo->GetBinContent(bin)	get content of bin
histo->GetBinError(bin)	get error of bin
histo->GetMinimum()	get minimum bin content
histo->GetMaximum()	get maximum bin content
histo->GetMinimumBin()	get bin in which minimum occurs
histo->GetMaximumBin()	get bin in which maximum occurs
histo->GetEntries()	get the number of entries
histo->GetSum()	gGet the sum of the bin contents
histo->Integral(bin1,bin2)	get the bin contents from bin1 to bin2 (incl.)
histo->GetMean()	get mean of x
histo->GetRMS()	Get RMS of x
....	....

Histogram Errors

By default, the **error** for a histogram bin is calculated as

$$\delta N = \sqrt{N}$$

with N the content of the bin. This is only correct if Poissonian statistics are applicable, *i.e.* if N represents *counts*. For *weighted* events, the error is given by

$$\delta N = \sqrt{\sum w_i}$$

This is has to be set explicitly, before filling the histogram:

histo->Sumw2()

You can also set the errors by hand:

histo->SetBinError(bin,error)

Manipulating Histogram Content

Action	**Example**
Group bins	histo->Rebin(2)
Add another histogram	histo->Add(otherhisto,1)
Add a function	histo->Add(afunction,-3.1415)
Add two histos	histo->Add(h1,h2,12,-11)
Divide with another histogram	histo->Divide(anotherhisto)
Multiply with another histogram	histo->Multiply(anotherhisto)
Scale with a factor	histo->Scale(100)
Smooth histogram	histo->Smooth(3)

More documentation

A lot more information and examples on histograms can be found in the USER MANUAL on

`ftp://root.cern.ch/root/doc/chapter3.pdf`

Also, on `http://root.cern.ch/root/Tutorials.html` check tutorials 6,7,8,12,13,24,32–34 for more histogramming examples.

Exercises

① Modify example 25. of the tutorial to display three full periods of a sin-wave (and get rid of the ugly brown background color). Make the marker a full square and change the line color to yellow.

② Experiment with the histogram drawing options, starting from example 24. of the tutorials.