

# ROOT basic tutorial

<https://root.cern.ch/>: installation, tutorials, class reference guide, ...

---

Ricardo Barrué, based on slides from prof. Patricia Muíño

LIP Summer Internships 2022



1. ROOT basics
2. ROOT classes and reference guide
3. GUI
4. cling interpreter
5. tutorial

# What is ROOT ?

Framework for data analysis based on C++.

- store/read data
- process data
- analyse data (histograms, statistical analysis, ...)

Developed, supported and widely used by the High Energy Physics community.

Three possible ways to use it:

- Graphical User Interface (GUI)
- C++ interpreter (cling)
- macros, programs, libraries (can be used as collection of libraries)

There are many classes in ROOT:

- TH1, TH2: histograms
- TFile: input/output files
- TCanvas: screens to plot graphical objects (histograms, graphs, ...)
- TTrees: save data in structured way
- many, many more...

No need to know all of them, can learn about different classes as necessary.

ROOT reference guide: <https://root.cern/reference/>

- tip: searching "(something you need) root cern" on Google tends to work very well

# ROOT class documentation: example

## TH1F Class Reference

Histogram Library

1-D histogram with a float per channel (see [TH1](#) documentation)

Definition at line 575 of file [TH1.h](#).

### Public Member Functions

#### TH1F ()

Constructor.

#### TH1F (const char \*name, const char \*title, Int\_t nbinsx, const Double\_t \*xbins)

Create a 1-Dim histogram with variable bins of type float (see [TH1::TH1](#) for explanation of parameters)

#### TH1F (const char \*name, const char \*title, Int\_t nbinsx, const Float\_t \*xbins)

Create a 1-Dim histogram with variable bins of type float (see [TH1::TH1](#) for explanation of parameters)

#### TH1F (const char \*name, const char \*title, Int\_t nbinsx, Double\_t xlow, Double\_t xup)

Create a 1-Dim histogram with fix bins of type float (see [TH1::TH1](#) for explanation of parameters)

#### TH1F (const TH1F &h1f)

Copy Constructor.

#### TH1F (const TVectorF &v)

Create a histogram from a TVectorF by default the histogram name is "TVectorF" and title = "".

**TH1F:** 1D histograms with floats

## TH2F Class Reference

Histogram Library

2-D histogram with a float per channel (see [TH1](#) documentation)

Definition at line 251 of file [TH2.h](#).

### Public Member Functions

#### TH2F ()

Constructor.

#### TH2F (const char \*name, const char \*title, Int\_t nbinsx, const Double\_t \*xbins, Int\_t nbinsy, const Double\_t \*ybins)

Constructor.

#### TH2F (const char \*name, const char \*title, Int\_t nbinsx, const Double\_t \*xbins, Int\_t nbinsy, Double\_t ylow, Double\_t yup)

Constructor.

#### TH2F (const char \*name, const char \*title, Int\_t nbinsx, const Float\_t \*xbins, Int\_t nbinsy, const Float\_t \*ybins)

Constructor.

#### TH2F (const char \*name, const char \*title, Int\_t nbinsx, Double\_t xlow, Double\_t xup, Int\_t nbinsy, const Double\_t \*ybins)

Constructor.

#### TH2F (const char \*name, const char \*title, Int\_t nbinsx, Double\_t xlow, Double\_t xup, Int\_t nbinsy, Double\_t ylow, Double\_t yup)

Constructor.

#### TH2F (const TH2F &h2f)

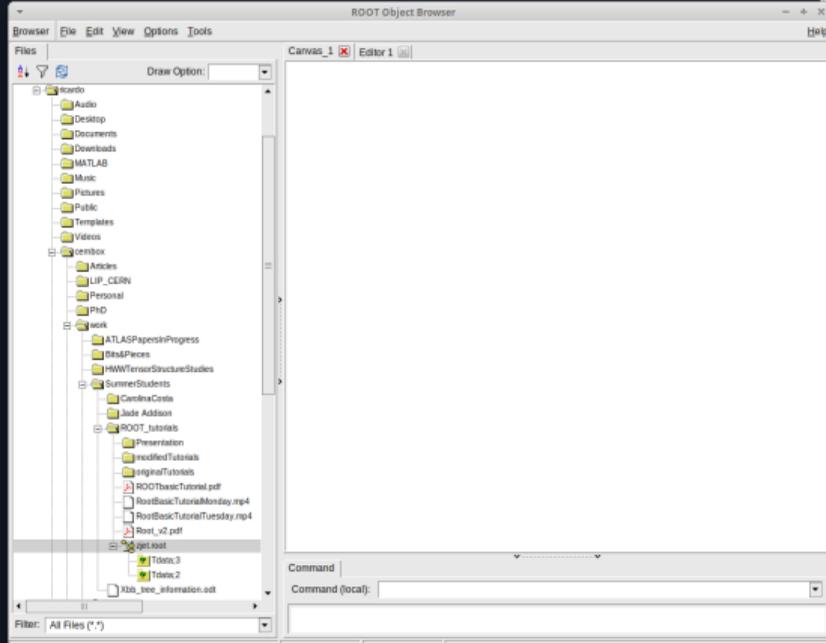
Copy constructor.

**TH2F:** 2D histograms with floats

# The GUI

Useful for browsing directories and files, accessed by creating a TBrowser.

```
powerhouse[~/cernbox/work/SummerStudents/ROOT_tutorials]$ root
root [0] TBrowser tb
(TBrowser &) Name: Browser Title: ROOT Object Browser
root [1] (TFile *) 0x55a6ba5fe1d0
```



# The GUI II - browsing files

The screenshot displays the ROOT Object Browser interface. On the left, a file browser shows a tree structure with folders like 'Pictures', 'Public', 'Templates', 'Videos', 'cerbox', 'work', and 'root'. The 'root' folder is expanded, showing subfolders like 'Tutorials' and 'Tutorials2'. The main canvas, titled 'pz.pz', shows a histogram of the variable 'pz.pz'. The x-axis ranges from -2000 to 2000, and the y-axis ranges from 0 to 70000. A statistics box in the top right corner provides the following data:

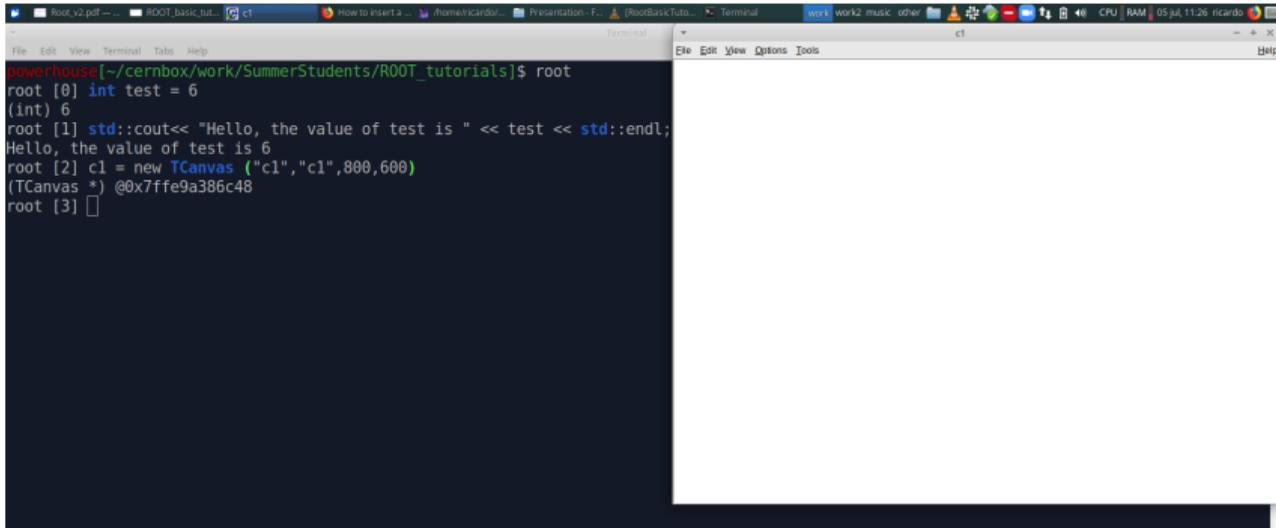
htemp	
Entries	1000000
Mean	-0.4084
Std Dev	341.7

At the bottom of the window, there are fields for 'Command' and 'Command (local):'. The status bar at the very bottom shows 'Pad graphics frame', 'TFrame', '570,648', and 'x=492.05, y=10070.7'.

# The cling interpreter

The most basic way to (really) use ROOT.

- can be used interactively or via macros/programs
- can execute instruction by instruction or compile programs



```
powerhouse[~/cernbox/work/SummerStudents/ROOT_tutorials]$ root
root [0] int test = 6
(int) 6
root [1] std::cout<< "Hello, the value of test is " << test << std::endl;
Hello, the value of test is 6
root [2] c1 = new TCanvas ("c1", "c1", 800, 600)
(TCanvas *) @0x7ffe9a386c48
root [3] □
```

**On to the actual tutorial...**

---

## Before we begin.

Whenever necessary: you can open these documentation pages in your browser (they will be tremendously helpful).

- [TFile](#)
- [TTree](#)
- [TH1F](#)
- [TH2F](#)
- [TCanvas](#)

## Exercise 2 - methods

The methods used in exercise 2 are the following:

- `TFile::Open()`: opens file in ROOT session
- `TFile::Get()`: loads object from file into memory (inherited from `TDirectoryFile` class)
- `TTree::Print()`: prints summary of tree contents
- `TTree::GetEntries()`: gets the number of entries in the tree
- `TTree::Draw()`: draws variable or expression for entries and objects that pass an (optional) selection.

## Exercise 3 - methods (already implemented)

The methods already implemented in exercise 3 are the following:

- TFile constructor
- TFile::Get()
- TH1F constructor (derived from TH1 constructor)
- TTree::Draw()

If you can't find a solution with this and the "raw" class documentation pages given in the beginning, jump to the next slide for some tips.

## Exercise 3 - tips

- 3b): see if the try the [TH2F](#) class is what you need
- 3c): see if the options of the [TTree::Draw\(\)](#) can help you
- 3d) and 3e): if you can't find the functions in the direct documentation of [TH1F](#), check the tabs "Public Member Functions Inherited from TH1" (for general 1D histograms) or the "Public Member Functions Inherited from TAttLine" (to change general line properties, such as color)
- 3e) if you want to see how/what formulas are defined in ROOT (to define fitting functions), check out the [TFormula](#) class
- if you want to see the different options to draw the histograms, check out the [THistPainter](#) class

## Exercise 4 - methods (already implemented)

The methods already implemented in exercise 4 are the following:

- `TFile` constructor
- `TFile::Get()`
- `TH1D` constructor (derived from `TH1` constructor)
- `TH1D::Fill()`: increment bin with abscissa  $X$  with a weight  $w$ .

For the C++ code only:

- `TTree::SetBranchAdress()`: links the program variables to the variables coming from the tree
- `TTree::GetEntries()`
- `TTree::GetEntry()`: loads the values from a given entry to the program variables

If you can't find a solution with this and the "raw" class documentation pages given in the beginning, jump to the next slide for some tips.

## Exercise 4 - tips

- 4a): if you can't find the Divide method in the direct documentation of [TCanvas](#), check the tab "Public Member Functions Inherited from TPad" (the base graphics drawing class)
- 4a): see if there is a method to change between pads similar to one you can use to change directories in the command line