Saving ROOT objects into a file

Anna Simon

November 6, 2015

Saving canvas into a file

- as graphics file.
- as ROOT file: saves an editable canvas, but the access to the objects in the canvas is limited. Useful to save a plot that might need to be updated.
- as C script: fully editable macro, recommended if frequent modifications of the plot are needed. Also useful to figure out the syntax for some graphical objects.
- as .tex file: using TIKZ package can be embedded in a Latex document.

Creating a ROOT file

```
1 TFile *fOut = TFile::Open("sample.root","RECREATE")
```

Options:

- NEW or CREATE: create a new file and open it for writing, if the file already exists the file is not opened.
- RECREATE: create a new file, if the file already exists it will be overwritten.
- UPDATE: open an existing file for writing. If no file exists, it is created.
- READ: open an existing file for reading (default).

NOTE: When opening a data file (.dat, .csv, etc.) use the fstream functions.

Saving histograms into a file

After the file is created, any new histogram will be created as an object within that file. Once the file is saved, the histograms are all saved within that file.

```
1 TFile *fOut = TFile::Open("sample.root","RECREATE")
2 TH1F *hist = new TH1F("hist","",100,0,10)
3 fOut->ls() // lists all the objects from the file
4
5 fOut->Write() //will save hist into sample.root
6 fOut->Close()
```

What is saved with a histogram

Add a latex object to the histogram, label the x-axis and change the histogram line color. Save the histogram to file.

To open a file:

```
1 TFile *fIn = TFile::Open("sample.root") //by default in read-only
      mode
2
3 fIn->ls()
4
5 hist->Draw()
```

Multiple files

NOTE: when multiple files are open the objects are attached to the last file used. To switch between files:

```
file1->cd()
```

- Create two files: file1.root and file2.root, use pointers file1 and file2, respectively.
- Create two histograms. hist1 (100 bins, 0 to 10, with a gaus(5,1), 10,000 events) that is appended to file1 and hist2 (100 bins, 0 to 10, uniform distribution, 10,000 events) that is appended to file2.
- Switch between the files and check that the histograms are where they are supposed to be.
- Save the files to disk.
- Open one of the files, examine its content, draw the histogram.

```
new TBrowser()
```

Object cycles

- Open file1.root in "UPDATE" mode, print the list of objects
- hist1->Fill(6,100)
- Save the file. Close ROOT
- Open file1.root, print the list of objects

When an object is modified, both the old version and the new one are preserved in the file. The newer the version of the object, the higher the cycle number.
To access a version of the object with a specific key value:

```
1 TH1* hSum4 //pointer that will point to the specified cycle
2 file->GetObject("hSum;4",hSum4) //name of the histogram is followed
      by the cycle number
```

The Write() function can be forced to overwrite the object:

```
1 myfile->Write(0,TObject::kOverwrite);
```

Accessing objects from multiple files

When multiple files are open, objects from other files are not directly accessible (error message: object is not defined in the current scope)
NOTE: the histograms have to be first drawn within their file scope to be accessible for further operations.

```
1 file1->cd()
2 TH1F * hPointer
3 file2->GetObject("hist2",hPointer) // creates a pointer to an object
      living in file2 that is accessible within file1 scope
4
5 //this will not work (hist2 is out of scope)
6 hist1->Add(hist2)
7
8 //this will work
9 hist1->Add(hPointer)
```

Add the two histograms from the previous example into one hSum. Save it into a sum.root file.

Saving other objects into a ROOT file

Objects other than histograms and trees do not live in the file scope and thus are not saved into a file automatically. They have to be explicitly saved into the file.

```
1 TFile *fOut = TFile::Open("canvas.root","RECREATE")
2 TCanvas *c1 = new TCanvas("c1","",800,600)
3
4 c1->Write() //saves c1 into the file
5
6 fOut->Write()
7 fOut->Close()
```

The same method is used for functions. When a fit to a histogram is saved into a file, all of its parameters are also preserved.

Exercise

- open sum.root file from previous exercise
- create a file named sumFit.root
- within the second file create a pointer to the hsum histogram
- create a fit function fSum that is a sum of gaus and linear function
- fit fSum to hSum
- save the histogram and fit into sumFit.root. Close ROOT
- open ROOT and sumFit.root file, examine its content, draw the histogram and the fit function, print out the fSum parameters to the screen.

Using ROOT libraries within a C++ code

```
1  #include <iostream>
2
3  //ROOT stuff
4  #include "TFile.h"
5  #include "TH1.h"
6
7  using namespace std;
8
9  int main(){
10
11 //create an output file and ROOT objects within it
12 TFile *file = TFile::Open("test.root","RECREATE");
13 TH1F * h1 = new TH1F("h1","",10,0,10);
14
15 //fill in the histogram
16 for (int i=0;i<10;i++)
17 {
18 h1->Fill(i,i*i);
19 }
20
21 //write the objects to a file and close it
22 file->Write();
23 file->Close();
24 }
```

How to compile it?

```
ROOTCFLAGS    := $(shell root-config --cflags)
ROOTLIBS      := $(shell root-config --libs)
ROOTGLIBS     := $(shell root-config --glibs)

CXX           = g++
CXXFLAGS      = -O2 -fPIC -g $(ROOTCFLAGS)

LD            = g++
LDFLAGS       = -O2
LIBS          = $(ROOTLIBS) $(SYSLIBS)
GLIBS         = $(ROOTGLIBS) $(SYSLIBS)

FILES         = analyzer.cpp
PROGRAMS      = analyzer
OBJS := $(FILES:.cpp=.o)

$(PROGRAMS):      $(OBJS)
$(LD) $(LDFLAGS) $^ $(LIBS) -o $@

%.o: %.cpp
$(CXX) $(CXXFLAGS) -c $< -o $@

clean:
@rm -f $(OBJS)
@rm -f $(PROGRAMS)
```

Make your first executable ROOT program

- copy the Makefile and test.C from /afs/crc.nd.edu/group/PHYS60070/makefile_example to your local folder
- compile and run the program, open the output file, inspect its content
- modify the test.C file to fit the histogram with a parabola. Make sure the function is saved into the output file!
- compile and run your new program, inspect the output file.