Data Trees & Ntuples

Create a data table of the function $f(x) = e^{-x} \sin^2(3x)$ for x between 0 and 10 in steps of 0.01

```
#include <TApplication.h>
#include <TNtuple.h>            //Root Ntuple header file
#include <math.h>

double func(double aX ){
  return exp( -aX ) * sin( 3.0 * aX ) * sin( 3.0 * aX);
}


int main( int argc, char **argv ){
  TApplication theApp( "App", &argc, argv );

  //Create an instance of a Tntuple object
  TNtuple ntuple("ntp1", "My First Ntuple", "x:y" );

  double xMin = -0.0;
  double xMax =  10.0;
  double deltaX = 0.01;

  for( double x = xMin; x <= xMax; x += deltaX )
    // Create the data table entry
    ntuple.Fill( x, func( x ) );

    // Create a plot of func(x) vs x
    ntuple.Draw("y:x");

  theApp.Run();
}
```

Saving Root Objects in a TFile

Save Root TNtuple to a Root TFile

```
#include <Tntuple.h>        // Root Ntuple header
#include <TFile.h>          // Root File Object header
#include <math.h>

double func(double aX ){
  return exp( -aX ) * sin( 3.0 * aX ) * sin( 3.0 * aX);
}

int main(){

  // Open a Root Data File Object
  TFile file("basic.root","RECREATE");

  //Create an instance of a TNtuple object
  TNtuple ntuple("ntp1", "My First Ntuple", "x:y" );

  double xMin = -0.0;
  double xMax =  10.0;
  double deltaX = 0.01;

  for( double x = xMin; x <= xMax; x += deltaX )
    // Create the data table entry
    ntuple.Fill( x, func( x ) );


  // Write all known objects to file
  file.Write();

  return 0;
}
```

Reading a TFile

Use ROOT Interpreter to study Ntuples

Start root!

In the ROOT shell type:

```
TFile file ("basic.root");

ntp1->Draw("y:x");
```

The Canvas

```
Creating: static Tcanvas cc("cc","Title",10,10,800,600);

Zoning: cc.Divide(2,2);

Navigation: cc.cd(3);

Drawing: ntp1->Draw("y:x");
```