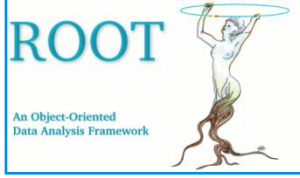
Εισαγωγή στο ROOT

- Το ROOT
- Ξεκινώντας το περιβάλλον του ROOT
- Απλές πράξεις
- Εκτελώντας αρχεία πηγαίου κώδικα
- Τύποι δεδομένων του ROOT
- Παραδείγματα
  - Τυχαίοι αριθμοί
  - Γράφημα συνάρτησης

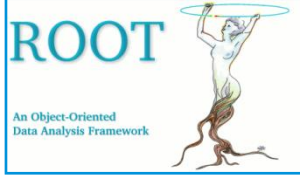
Το ROOT



- Το ROOT είναι ένα αντικειμενοστραφές πρόγραμμα και βιβλιοθήκη το οποίο αναπτύχθηκε στο CERN.
- Παρ’ ότι αρχικά αναπτύχθηκε για της ανάγκες της ανάλυσης δεδομένων της σωματιδιακής φυσικής σήμερα έχει καθιερωθεί ως ένα πανίσχυρο εργαλείο για επεξεργασία δεδομένων (data mining).
- Η ανάπτυξή του ξεκίνησε αρχικά από τους Rene Brune και Fons Rademakers το 1994. Το ROOT αναπτύσσεται σε C++.
- Μέσα στα εργαλεία που περιλαμβάνει είναι και τα ακόλουθα:
  - Γραφικό περιβάλλον χρήστη (Graphical User Interface)
  - Πολλές έτοιμες κλάσεις (Container Classes)
  - Μεταφραστής εντολών γραμμής C++ (Command Line Interpreter)
  - Εργαλεία διαχείρισης δεδομένων

  και πολλά άλλα.
- Στο ROOT βασίζεται όλη η ανάλυση των δεδομένων των πειραμάτων του Μεγάλου Αδρονικού Επιταχυντή Συγκρουομένων Δεσμών (Large Hadron Collider) του CERN. Ο όγκος των δεδομένων αυτών των πειραμάτων υπολογίζεται σε δεκάδες Penta Bytes (10^{15} Bytes) ανά έτος.

Ξεκινώντας το περιβάλλον του ROOT

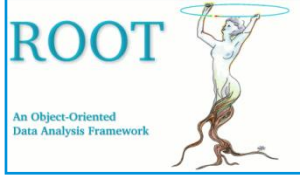


- Στη γραμμή εντολών του υπολογιστή σας εκτελέστε root για να ξεκινήσει το περιβάλλον.

```
[panos@pc-247 Root]$
[panos@pc-247 Root]$ root
  ****************************************
  *                                      *
  *        W E L C O M E  to  R O O T     *
  *                                      *
  *   Version   5.13/02    29 August 2006  *
  *                                      *
  *  You are welcome to visit our Web site  *
  *          http://root.cern.ch         *
  *                                      *
  ****************************************

FreeType Engine v2.1.9 used to render TrueType fonts.
Compiled on 29 August 2006 for linux with thread support.

CINT/ROOT C/C++ Interpreter version 5.16.14, August 18, 2006
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0]
root [0]
root [0] .q
[panos@pc-247 Root]$
```

- Για να τερματιστεί το περιβάλλον εκτελέστε το .q όπως παραπάνω.

Απλές πράξεις



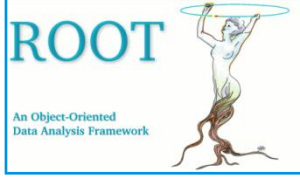
- Χρησιμοποιώντας την γραμμή εντολών μπορούμε να κάνουμε απλές πράξεις όπως παρακάτω.

```
root [0] 1+sqrt(16)
(const double)5.00000000000000000e+00
root [1]
root [1] cos(0.4)+sin(1.2)
(double)1.85310007997011139e+00
root [2]
root [2] for(int i=0; i<5; ++i) cout << "Hello " << i << endl;
Hello 0
Hello 1
Hello 2
Hello 3
Hello 4
root [3] .q
[panos@pc-247 Root]$
```

- Χρησιμοποιώντας τη γραμμή εντολών μπορούμε να εκτελέσουμε εντολές πολλών γραμμών (Προσοχή στις αγκύλες {}).

```
root [0] {
end with '}', '@':abort > int j=0;
end with '}', '@':abort > for (int i=0; i<3; ++i)
end with '}', '@':abort > {
end with '}', '@':abort > j=j+i;
end with '}', '@':abort > cout << "i=" << i << ",  j=" << j << endl;
end with '}', '@':abort > }
end with '}', '@':abort > }
i=0,  j=0
i=1,  j=1
i=2,  j=3
root [1]
```

Εκτελώντας αρχεία πηγαίου κώδικα

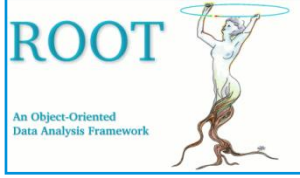


- Ο καλύτερος τρόπος ανάπτυξης προγραμμάτων είναι σε αρχεία πηγαίου κώδικα.
- Η εκτέλεσή τους γίνεται από την γραμμή εντολών .x filename.C .

```
{

  for(int i=0; i<10; ++i){
     double a = sqrt(i);
     double b = pow(i,2);
     cout << "i=" << i  << "  pow(i,2)=" << b << "  sqrt(i)=" << a << endl;
  }

}
```

```
root [0]
root [0] .x script1.C
i=0  pow(i,2)=0  sqrt(i)=0
i=1  pow(i,2)=1  sqrt(i)=1
i=2  pow(i,2)=4  sqrt(i)=1.41421
i=3  pow(i,2)=9  sqrt(i)=1.73205
i=4  pow(i,2)=16  sqrt(i)=2
i=5  pow(i,2)=25  sqrt(i)=2.23607
i=6  pow(i,2)=36  sqrt(i)=2.44949
i=7  pow(i,2)=49  sqrt(i)=2.64575
i=8  pow(i,2)=64  sqrt(i)=2.82843
i=9  pow(i,2)=81  sqrt(i)=3
root [1] .q
[panos@pc-247 Root]$
```

Τύποι δεδομένων του ROOT



- Οι ακόλουθοι τύποι δεδομένων ορίζονται στο ROOT και είναι ανεξάρτητοι από τον τύπο του υπολογιστή.
  - **Char_t** Signed Character 1 byte
  - **UChar_t** Unsigned Character 1 byte
  - **Short_t** Signed Short integer 2 bytes
  - **UShort_t** Unsigned Short integer 2 bytes
  - **Int_t** Signed integer 4 bytes
  - **UInt_t** Unsigned integer 4 bytes
  - **Long64_t** Portable signed long integer 8 bytes
  - **ULong64_t** Portable unsigned long integer 8 bytes
  - **Float_t** Float 4 bytes
  - **Double_t** Float 8 bytes
  - **Double32_t** Double 8 bytes in memory, written as a Float 4 bytes
  - **Bool_t** Boolean (0=false, 1=true)

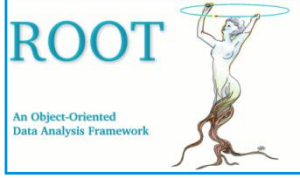
Παράδειγμα 1: Τυχαίοι Αριθμοί

ROOT
An Object-Oriented
Data Analysis Framework

- Το gRandom είναι δείκτης προς τον γεννήτορα τυχαίων αριθμών TRandom3 ("Γεννήτορας Mersenne-Twister" περίοδος 10^{600}).
- Μέσω του gRandom είναι δυνατή η παραγωγή των ακόλουθων κατανομών τυχαίων αριθμών
  - Rndm() or Uniform(min,max)
  - Gaus(mean,sigma)
  - Exp(tau)
  - BreitWigner(mean,sigma)
  - Landau(mean,sigma)
  - Poisson(mean)
  - Binomial(ntot,prob)
- Υπάρχει δυνατότητα επιλογής του γεννήτορα τυχαίων αριθμών σβήνοντας το gRandom και επαναδημιουργώντας το με τον κατάλληλο γεννήτορα ως εξής:

```
delete gRandom;
gRandom = new TRandom(1234)     // seed 1234
```

Παράδειγμα 1: Τυχαίοι Αριθμοί

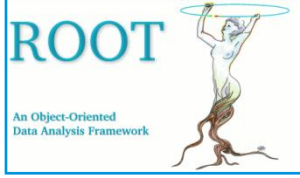


- Στο ακόλουθο παράδειγμα τυπώνουμε 10 τυχαίους αριθμούς από το 0 έως το 1, από το 100 έως το 200 και με κατανομή Gauss με μέσο 10 και σίγμα 2.

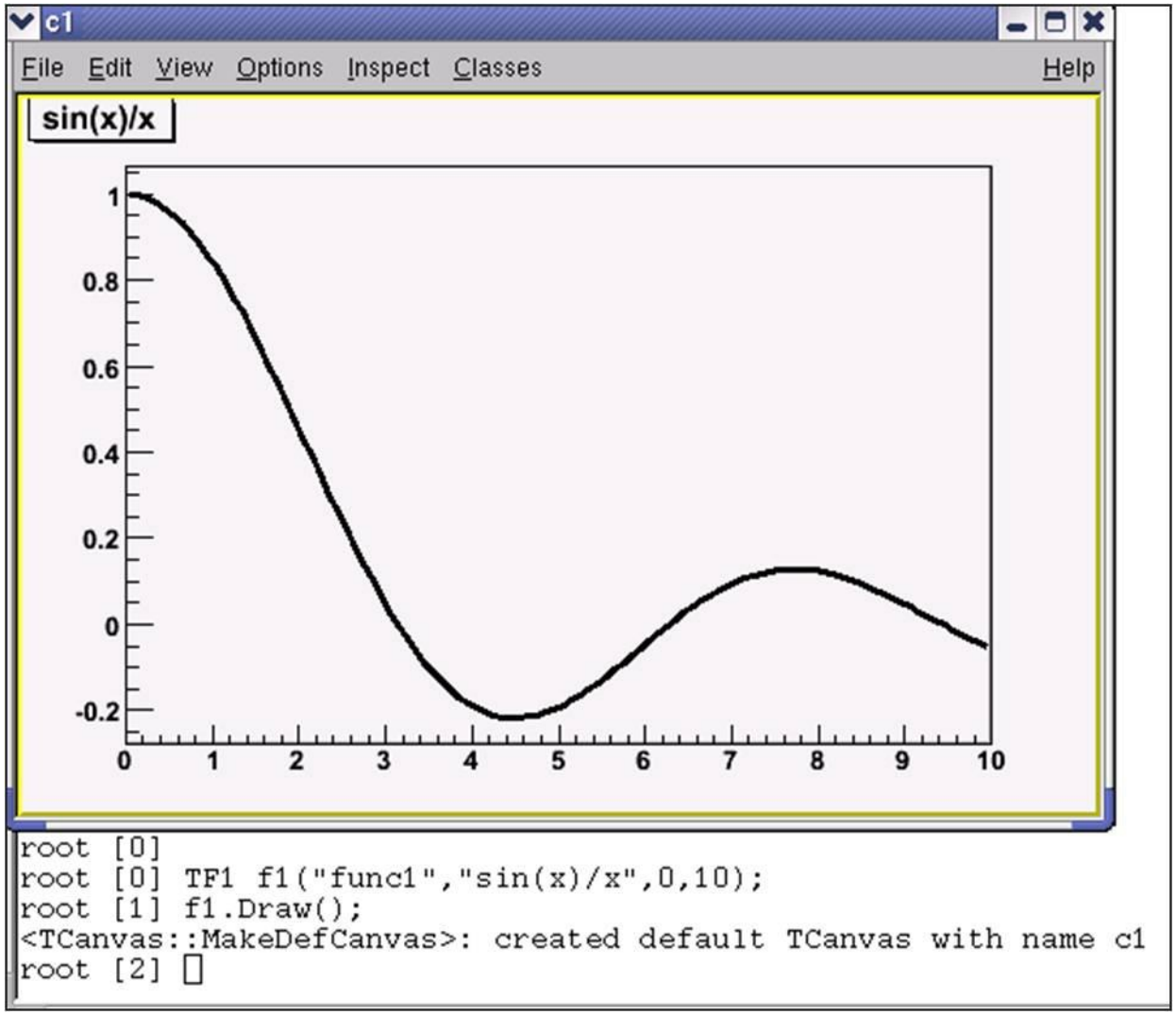
```
{
  delete gRandom;
  gRandom =  new TRandom(123456);    // Setting the seed

  int i;
  for(i=0; i<10; ++i){
    cout << "i=" << i;
    cout << "  Rndm():" << gRandom->Rndm();
    cout << " \tUniform(100,200):" << gRandom->Uniform(100,200);
    cout << " \tGaus(10,2):" << gRandom->Gaus(10,2) << endl;
  }

}
```

```
root [0] .x rndm_numbers.C
i=0  Rndm():0.634675    Uniform(100,200):127.528        Gaus(10,2):14.3087
i=1  Rndm():0.249985    Uniform(100,200):117.846        Gaus(10,2):10.1254
i=2  Rndm():0.98599     Uniform(100,200):164.014        Gaus(10,2):6.43961
i=3  Rndm():0.0506745   Uniform(100,200):117.655        Gaus(10,2):13.9345
i=4  Rndm():0.782631    Uniform(100,200):151.285        Gaus(10,2):5.72076
i=5  Rndm():0.0552331   Uniform(100,200):120.06         Gaus(10,2):8.92759
i=6  Rndm():0.67886     Uniform(100,200):102.248        Gaus(10,2):12.0665
i=7  Rndm():0.269913    Uniform(100,200):127.779        Gaus(10,2):7.6895
i=8  Rndm():0.55538     Uniform(100,200):154.261        Gaus(10,2):12.4892
i=9  Rndm():0.581703    Uniform(100,200):199.822        Gaus(10,2):10.3084
root [1] .q
[panos@pc-247 Root]$
```

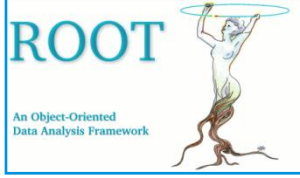
Παράδειγμα 2: Γράφημα συνάρτησης

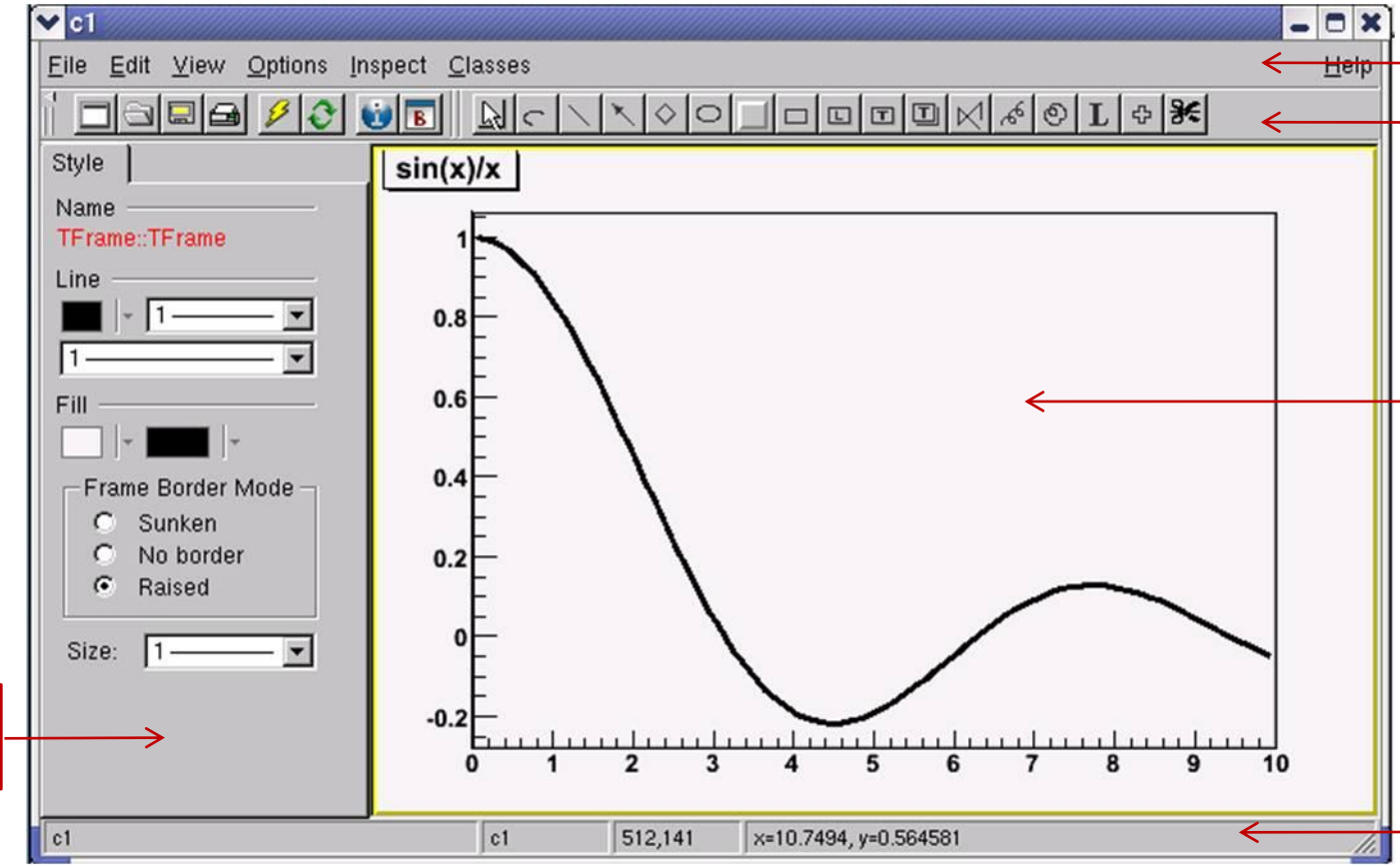


- Χρησιμοποιώντας την κλάση συναρτήσεων TF1 μπορούμε εύκολα να κάνουμε το γράφημα μιας συνάρτησης, όπως στο παρακάτω παράδειγμα.



Χρησιμοποιώντας το Γραφικό Περιβάλλον Χρήστη





- Menu bar – περιέχει τα βασικά μενού για διαχείριση files, print, clear canvas, inspect, κτλ.
- Tool bar – κουμπιά για διαχείριση σχεδιαστικών αντικειμένων όπως βέλη, κύκλοι κτλ.
- Canvas – επιφάνεια σχεδίασης αντικειμένων.
- Status bar – αναγραφή στοιχείων των επιλεγμένων αντικειμένων.
- Editor frame – δίνει την δυνατότητα στον χρήστη αλλαγής των χαρακτηριστικών των αντικειμένων στον καμβά.