

ROOT

warsztaty

Część 1

Strony internetowe

- **instrukcja instalacji dla różnych dystrybucji Linuxa**

root.cern/install/#download-a-pre-compiled-binary-distribution

- **pliki, których dziś będziemy używać:**

<https://github.com/KrzysztofProscinski/ROOT>

- **szerszy poradnik ROOT-a**

<https://www.fuw.edu.pl/~kpias/>

Dydaktyka -> Computer Tools for Nuclear Physics

Linux - powtórzenie

Komendy te należy wpisywać w linuxowy terminal.

> mkdir [folder]

- utworzenie nowego folderu

> touch [plik]

- utworzenie nowego pliku

> cd [folder]

- przejście do danego folderu

> cd ..

- przejście do folderu macierzystego

> cd

- przejście do folderu domowego

> ls

- zawartość obecnego folderu

> mv [plik] [folder]

- przeniesienie pliku do folderu

> mv [plik1] [plik2]

- zmiana nazwy pliku

> cp [plik] [folder]/

- skopiowanie pliku do folderu

> cp -r [folder1] [folder2]/

- skopiowanie folderu do folderu

> rm [plik]

- usunięcie pliku

Komendy do terminala będą oznaczane jako "> [treść komendy]". Przykładowo "> cd" oznacza, że w terminalu należy wpisać "cd".

"-r" trzeba dodawać zawsze do operacji na folderach.

.bashrc

.bashrc to plik domyślnie istniejący w Linuxach, do którego należy dopisać poniższe linijki, aby ROOT poprawnie działał.

> cd

> nano ~/.bashrc

“nano” to przykładowy edytor tekstowy w Linuxie. Zamiast tego można użyć innego edytora.

wpisać:

“export ROOTSYS=\$HOME/root

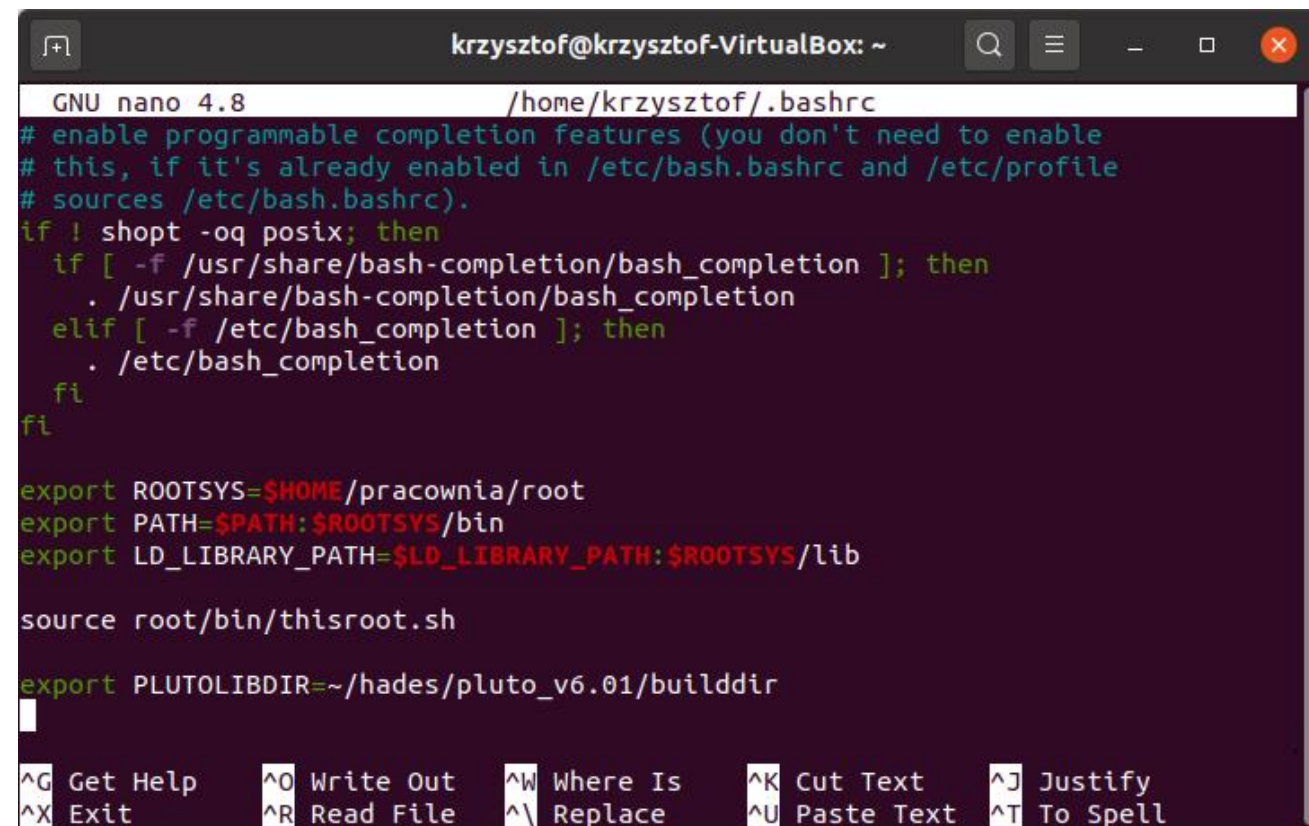
export PATH=\$PATH:\$ROOTSYS/bin

export

LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:
\$ROOTSYS/lib

source root/bin/thisroot.sh”

Niebieskie teksty w cudzysłowach oznaczają treści makr.



```
GNU nano 4.8 /home/krzysztof/.bashrc
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export ROOTSYS=$HOME/pracownia/root
export PATH=$PATH:$ROOTSYS/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ROOTSYS/lib

source root/bin/thisroot.sh

export PLUTOLIBDIR=~/.hades/pluto_v6.01/buildidir
```

Jak uruchomić ROOT-a

- **uruchamianie**

> root

- **opcje uruchamiania**

> root -l -b

“-l” - bez ekranu powitalnego

“-b” - bez grafiki

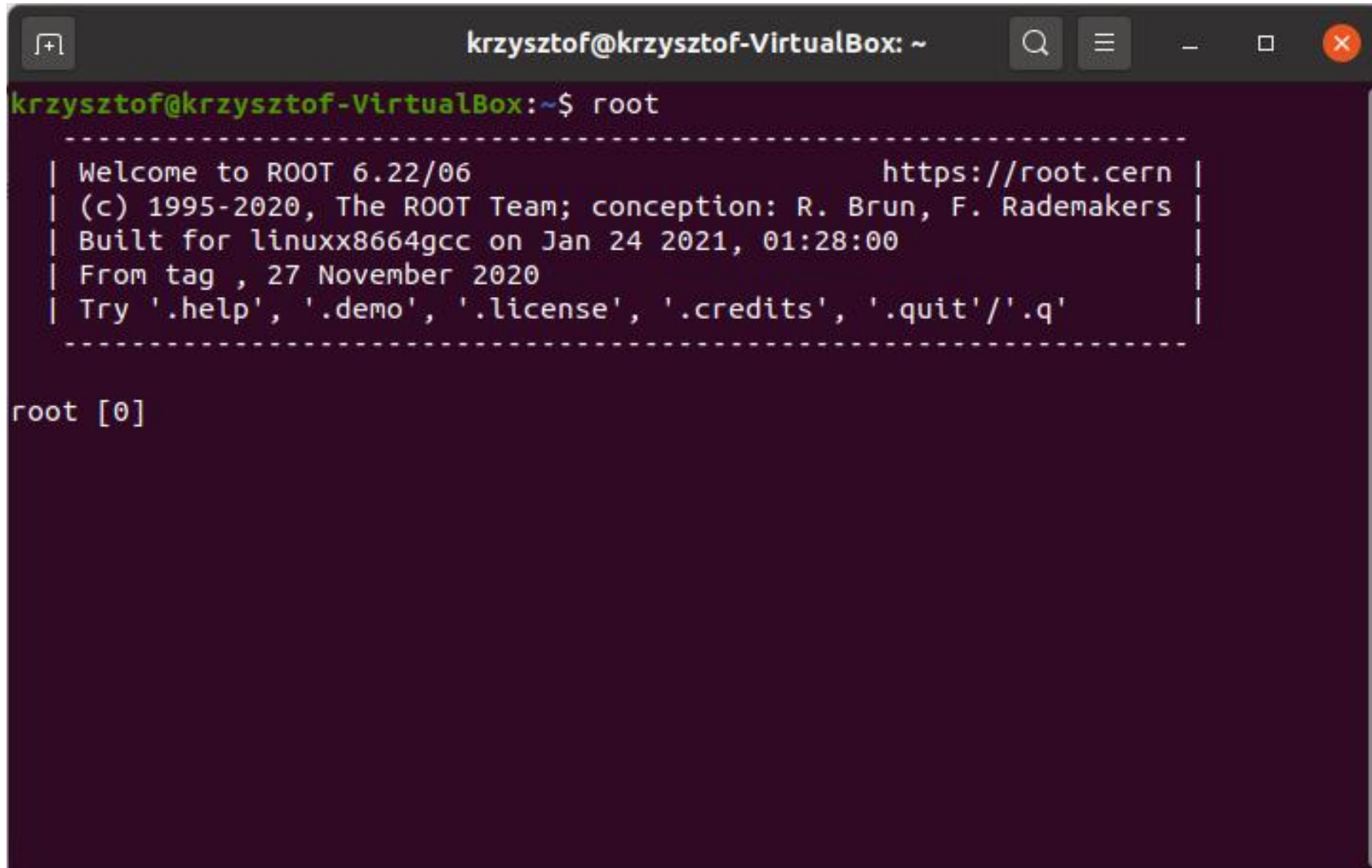
- **zamykanie**

> .q

Opcje można ze sobą składać, tzn. jeżeli zostaną wpisane obie, to obie zostaną uwzględnione.

Po uruchomieniu ROOT-a normalne linuxowe komendy (np. zmiana katalogu) są niedostępne. Aby móc ich użyć należy zamknąć ROOT-a.

Jak uruchomić ROOT-a



```
krzysztof@krzysztof-VirtualBox: ~  
krzysztof@krzysztof-VirtualBox:~$ root  
-----  
| Welcome to ROOT 6.22/06                                     https://root.cern |  
| (c) 1995-2020, The ROOT Team; conception: R. Brun, F. Rademakers |  
| Built for linuxx86_64gcc on Jan 24 2021, 01:28:00             |  
| From tag , 27 November 2020                                  |  
| Try '.help', '.demo', '.license', '.credits', '.quit'/'.q'   |  
-----  
root [0]
```

Zdalne połączenie (Windows)

- **wymagane programy**

PuTTY (lub inna aplikacja do zdalnego logowania)

Xming (lub inny serwer systemu X Windows)

- **uruchamianie**

uruchomić PuTTY

wpisać odpowiedni adres (lub adres IP) w pole "Host Name"

wcisnąć "Open"

- **dostęp do aplikacji graficznych**

uruchomić Xming, a następnie PuTTY

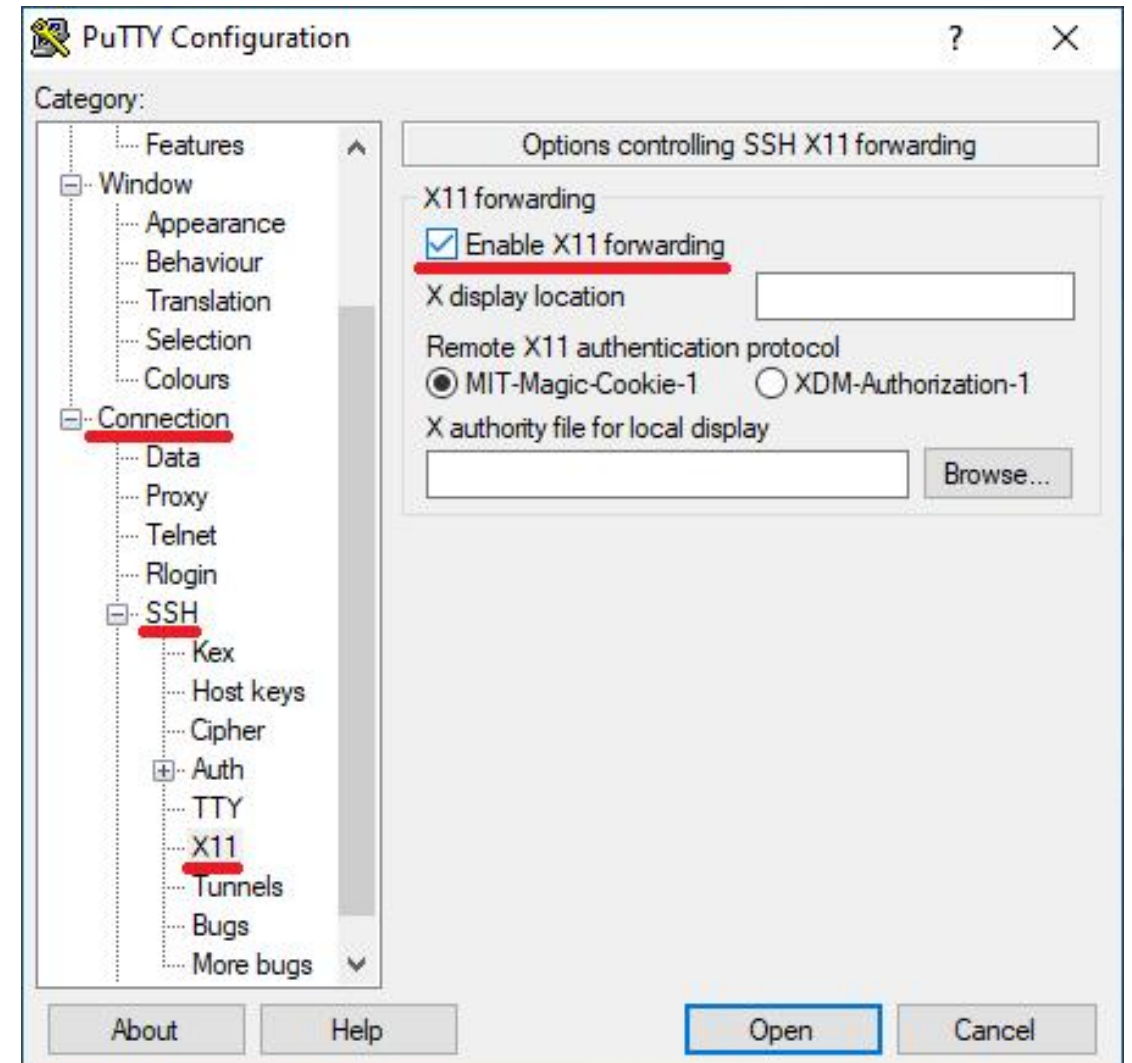
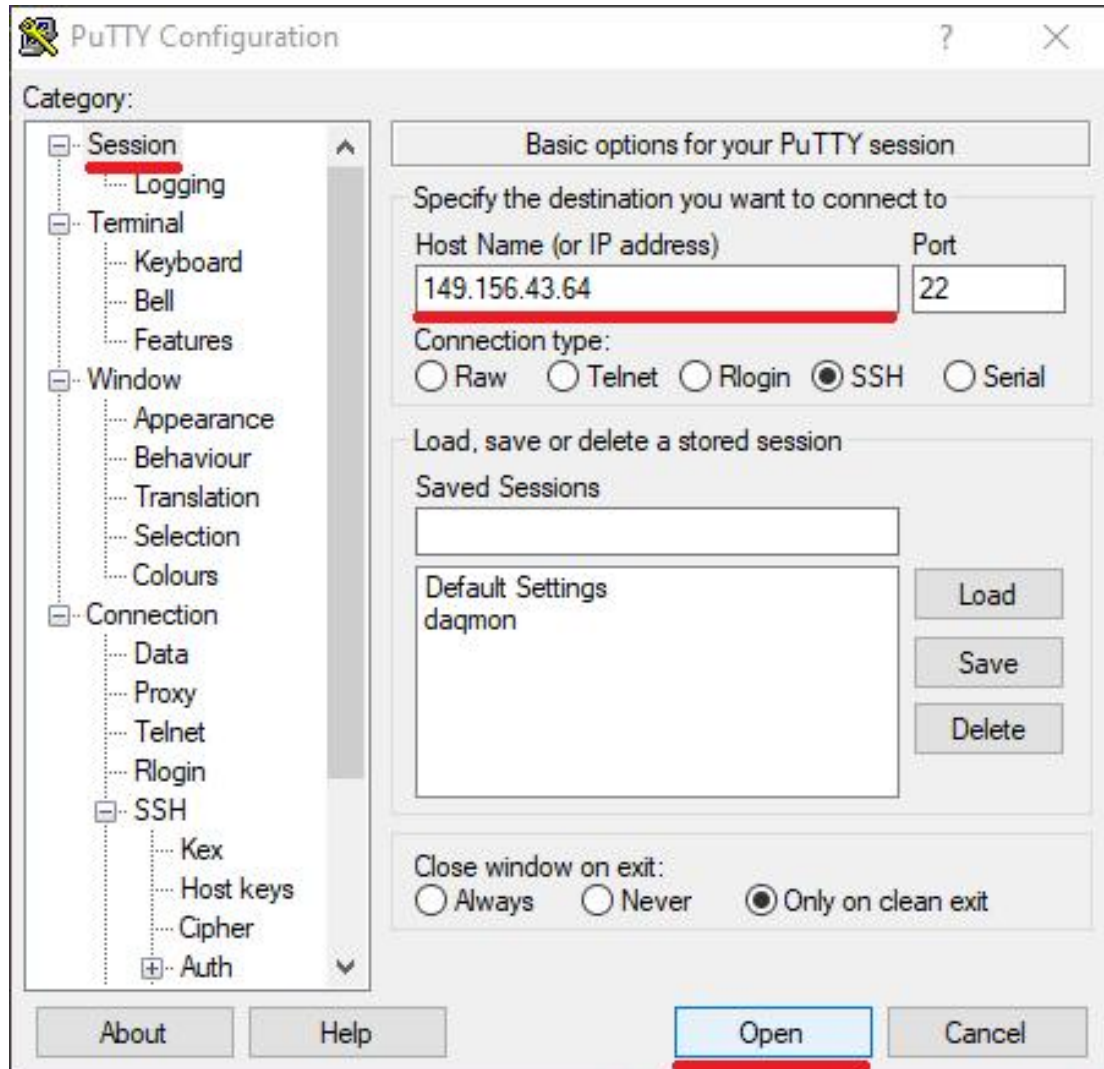
włączyć opcję "Enable X11 forwarding" (Connection->SSH->X11)

dalej jak przy normalnym uruchamianiu

Przykładowo adres Studenckiej Pracowni Komputerowej to `spk-ssh.if.uj.edu.pl`, a adres IP to `149.156.43.64`.

Na niektórych zdalnych komputerach użycie aplikacji graficznych może nie być możliwe i pozostaje jedynie korzystanie z terminala.

Zdalne połączenie (Windows)



Zdalne połączenie (Linux)

- **połączenie**

> ssh [login]@[serwer]

przykład

> ssh kproscin@149.156.43.64

- **kopiowanie plików między komputerami**

> scp kproscin@149.156.43.64:~/plik.txt .

> scp plik.txt kproscin@149.156.43.64:~/.

- **zamykanie połączenia**

> exit

Do kopiowania plików należy wpisać "scp [adres pliku] [adres docelowy]".

W pierwszym przypadku kopiowany jest plik "plik.txt" z komputera pracowni na komputer użytkownika. Kropka wpisana zamiast adresu docelowego oznacza, że adresem docelowym jest aktualnie otwarty w terminalu folder.

W drugim przypadku kopiowany jest plik z komputera użytkownika na komputer pracowni. ~/. oznacza, że adresem docelowym jest folder domowy.

Polecenia scp należy wpisać na komputerze użytkownika, nie na komputerze zdalnym.

rootlogon.C

- makro ładowane przy każdym uruchomieniu ROOT-a

> nano rootlogon.C

wpisać

```
"#include<iostream>
```

```
Bool_t rootlogon(void){
```

```
cout<<"hello"<<endl;
```

```
return kTRUE;
```

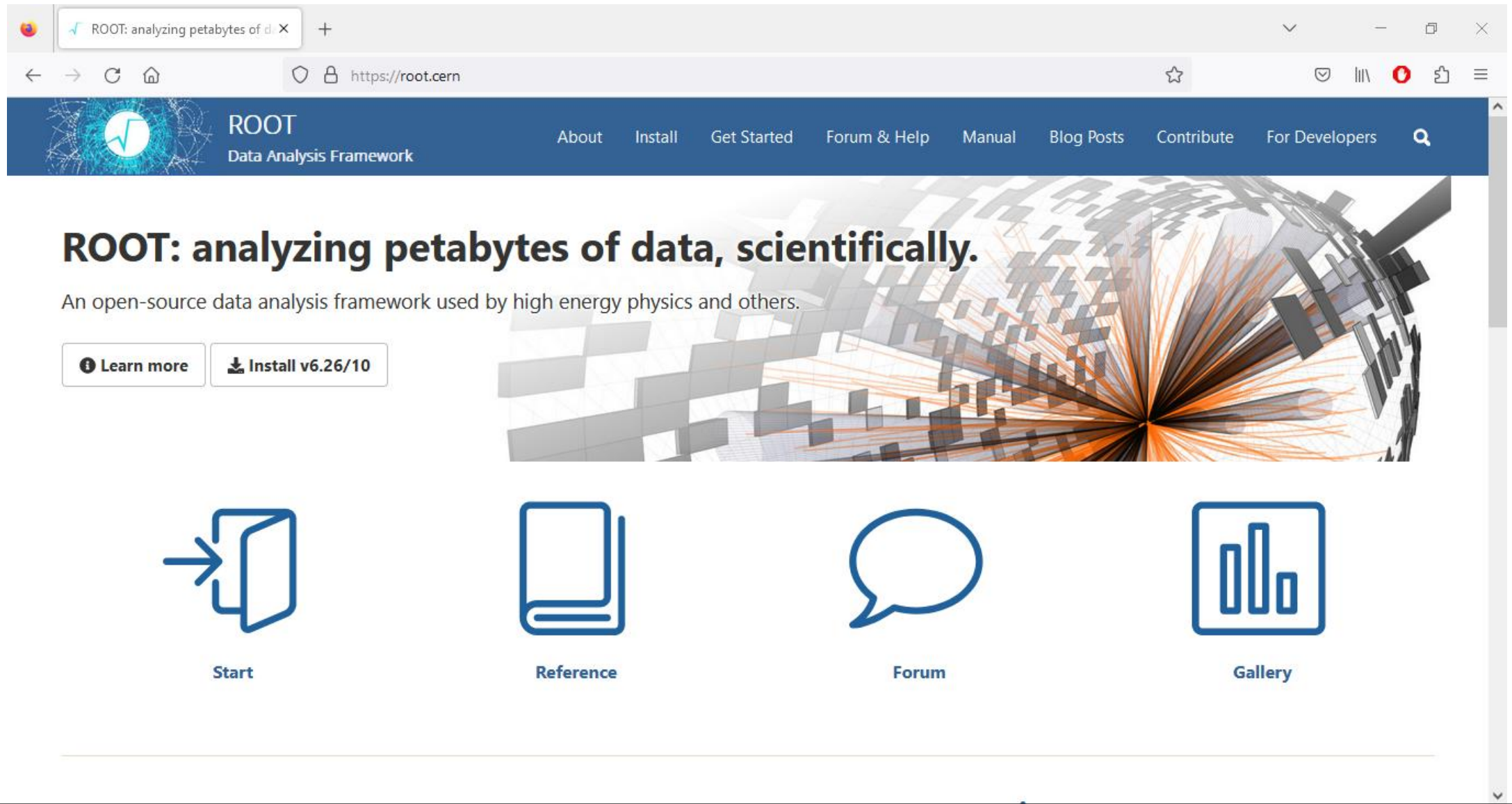
```
}"
```

Makro to służy do wykonywania dodatkowych procesów podczas uruchamiania ROOT-a, np. ładowania dodatkowych bibliotek.

Przy uruchamianiu ROOT-a automatycznie uruchamiane jest makro rootlogon.C, które znajduje się w aktualnie otwartym folderze. Do innych folderów należy utworzyć nowe makra rootlogon.C.

W przypadku takiego makra jak to po lewej, przy każdym uruchomieniu ROOT-a napisany zostanie tekst "hello".

Internetowa dokumentacja: root.cern



The image is a screenshot of a web browser displaying the ROOT Data Analysis Framework website. The browser's address bar shows the URL `https://root.cern`. The website has a dark blue header with the ROOT logo (a stylized 'R' with a checkmark) and the text 'ROOT Data Analysis Framework'. To the right of the logo, there is a navigation menu with links: 'About', 'Install', 'Get Started', 'Forum & Help', 'Manual', 'Blog Posts', 'Contribute', and 'For Developers'. Below the header, the main content area features a large, abstract graphic of a particle detector with orange and grey elements. The text 'ROOT: analyzing petabytes of data, scientifically.' is prominently displayed, followed by the subtitle 'An open-source data analysis framework used by high energy physics and others.' Below this, there are two buttons: 'Learn more' and 'Install v6.26/10'. At the bottom of the page, there are four large, blue-outlined icons representing different sections: 'Start' (a document with an arrow), 'Reference' (a book), 'Forum' (a speech bubble), and 'Gallery' (a bar chart).

ROOT: analyzing petabytes of data, scientifically.

An open-source data analysis framework used by high energy physics and others.

[Learn more](#) [Install v6.26/10](#)

[Start](#) [Reference](#) [Forum](#) [Gallery](#)

Internetowa dokumentacja: root.cern

The screenshot shows a web browser window displaying the ROOT TH2F Class Reference page. The browser's address bar shows the URL `https://root.cern.ch/doc/master/classTH2F.html`. The page header features the ROOT logo, the text "ROOT Reference Guide", a "Version master" dropdown, and a search bar. A left sidebar contains a "Histogram Library" with a tree view of classes, where "TH2F" is selected. The main content area is titled "TH2F Class Reference" and includes a breadcrumb "Histogram Library » Histogram classes.". It describes the "2-D histogram with a float per channel (see TH1 documentation)" and notes its "Definition at line 257 of file TH2.h.". Below this, the "Public Member Functions" section lists several constructors for the TH2F class, each with its signature and a link to "More..." for details.

ROOT: TH2F Class Reference

Version master

Search

Histogram Library

- Painting classes
- Histogram classes.
 - TAxis
 - TH1
 - TH1C
 - TH1D
 - TH1F
 - TH1I
 - TH1S
 - TH2C
 - TH2D
 - TH2F**
 - TH2I
 - TH2Poly
 - TH2PolyBin
 - TH2S
 - TH3
 - TH3C
 - TH3D
 - TH3F
 - TH3I
 - TH3S

TH2F Class Reference

Histogram Library » Histogram classes.

List of all members | Public Member Functions | Static Public Member Functions | Protected Member Functions | Friends | List of all members

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

Definition at line **257** of file **TH2.h**.

Public Member Functions

TH2F ()
Constructor. [More...](#)

TH2F (const char *name, const char *title, **Int_t** nbinsx, const **Double_t** *xbins, **Int_t** nbinsy, const **Double_t** *ybins)
Constructor (see [TH2::TH2](#) for explanation of parameters) [More...](#)

TH2F (const char *name, const char *title, **Int_t** nbinsx, const **Double_t** *xbins, **Int_t** nbinsy, **Double_t** ylow, **Double_t** yup)
Constructor (see [TH2::TH2](#) for explanation of parameters) [More...](#)

TH2F (const char *name, const char *title, **Int_t** nbinsx, const **Float_t** *xbins, **Int_t** nbinsy, const **Float_t** *ybins)
Constructor (see [TH2::TH2](#) for explanation of parameters) [More...](#)

TH2F (const char *name, const char *title, **Int_t** nbinsx, **Double_t** xlow, **Double_t** xup, **Int_t** nbinsy, const **Double_t** *ybins)
Constructor (see [TH2::TH2](#) for explanation of parameters) [More...](#)

TH2F

ROOT master - Reference Guide Generated on Mon Feb 27 2023 09:47:54 (GVA Time) using Doxygen 1.9.5

Proste obliczenia matematyczne

- **co działa, co nie działa**

> 2+3 - dodawanie

> 2*3 - mnożenie

> 2^3 - dodawanie!

Poprawny zapis takiego potęgowania to "TMath::Pow(2,3)"

- **TMath**

Wszystkie klasy w ROOT-cie zaczynają się od "T".

> TMath::Sqrt(4) - pierwiastek

> TMath::Pi() - liczba pi

> TMath::Sin(0) - sinus

root.cern.ch/root/html524/TMath.html

Na tej stronie znajduje się spis wszystkich funkcji matematycznych w klasie TMath.

Typy zmiennych

Wszystkie typy zmiennych w ROOT-cie kończą się “_t”.

Char_t	- char (znak)
Short_t	- short integer (liczba całkowita)
Int_t	- integer (liczba całkowita)
Long64_t	- long64 (liczba całkowita)
Float_t	- float (liczba zmiennoprzecinkowa)
Double_t	- double (float podwójnej precyzji)
Bool_t	- boolean (zmienna boolowska)

Stałe w ROOT-cie zaczynają się od “k”. Przykładowo klasa Bool_t zawiera dwie stałe: “kTRUE” oraz “kFALSE”.

Niektóre typy zmiennych posiadają wersję “unsigned”, czyli pozbawioną informacji o znaku (+ lub -). Przykładowo Short_t posiada wersję UShort_t. Obie wersje posiadają 16 bitów, więc Short_t obejmuje zakres od -32768 do 32767, natomiast UShort_t obejmuje zakres od 0 do 65535. Istnieją jeszcze UChar_t, UInt_t oraz ULong64_t.

Makra

macro.C

```
"#include <iostream>  
using namespace std;
```

```
Int_t macro(){  
    for(Int_t i=0; i<10; i++){  
        cout<<i<<endl;  
    }  
    return 0;  
}"
```

A screenshot of a C++ IDE window titled 'macro.C' with the path '~/hades/doc'. The window contains a C++ program. The code is as follows:

```
1 #include <iostream>  
2 using namespace std;  
3  
4 Int_t macro(){  
5     for(Int_t i=0; i<10; i++){  
6         cout<<"i"<<endl;  
7     }  
8     return 0;  
9 }
```

The line number 8 is highlighted. The IDE interface includes a top bar with 'Open', 'Save', and window control buttons. The bottom status bar shows 'C++', 'Tab Width: 8', 'Ln 8, Col 1', and 'INS'.

Makra

- **utworzenie i edycja makra**

- > touch macro.C

- > nano macro.C

- **uruchamianie**

- > root

- > .L macro.C //kompilacja (opcjonalna)

- > .x macro.C //wykonywanie makra

W ROOT-cie kompilacja nie jest konieczna.
Można wykonać makro od razu.

input, output

- **output**

```
"cout<<"hello"<<endl;"
```

otrzymamy napis hello

```
"Int_t t=0;"
```

```
cout<<t<<endl;"
```

otrzymamy wartość t, czyli 0

- **input**

```
"Int_t s;"
```

```
cin >> s;"
```

ustawiamy wartość s na to co wpiszemy z klawiatury

Istnieją pewne znaki specjalne, takie jak np. tabulator, które należy wpisywać w następujący sposób: \[znak]

"\a" - alert/bell

"\n" - newline

"\t" - tab

"\b" - backspace

"\r" - return to left margin

Znaki specjalne muszą znajdować się wewnątrz cudzysłowa.

Get, Set

- Set - ustawianie, np.:

```
"TH1F *hist = new TH1F("h1","Title",10,0.,5.);  
hist.SetTitle("nowy tytuł");  
hist.SetMinimum(0);  
hist.SetMaximum(10);"
```

- Get - pobieranie, np.:

```
"TH1F *hist = new TH1F("h1","Title",10,0.,5.);  
hist.GetBinContent(10);  
hist.GetBinError(10);"
```

Ustawimy kolejno:

- tytuł histogramu
- dolny histogramu
- górny kres histogramu

Otrzymamy wartości kolejno:

- liczby zliczeń w dziesiątym binie histogramu
- niepewności liczby zliczeń w dziesiątym binie histogramu

Obsługa plików

- **TFile**

Definicja pliku z opcją "RECREATE".

```
"TFile* f = new TFile ("file.root", "RECREATE");  
TH1F *hist = new TH1F("h1","Title",200,-1.,1.);  
[...]
```

```
f->cd();
```

```
hist->Write();
```

Zapisywanie do pliku.

```
f->Close();"
```

Zamykanie pliku.

- **pobranie obiektu z pliku**

```
"TFile* f = new TFile ("file.root");  
TH1F *hist2= (TH1F*)f -> Get("h1");"
```

W tym przypadku plik file.root musi istnieć już wcześniej i zawierać histogram o nazwie "h1".

- **Biblioteki**

<iostream> - input/output na ekran

<ifstream> - input do pliku

<ofstream> - output do pliku

<fstream> - input/output do pliku

- **Opcje do wpisania przy definiowaniu pliku:**

CREATE, NEW, READ, RECREATE,
UPDATE

Obsługa plików - opcje

Opcja	Krótki opis	Jeśli plik o takiej nazwie wcześniej nie istniał	Jeśli plik o takiej nazwie wcześniej już istniał
CREATE	nowy plik	Utworzony zostaje nowy plik	Stary plik nie zostaje otwarty, a dane nie są nigdzie zapisywane
NEW	nowy plik	Utworzony zostaje nowy plik	Stary plik nie zostaje otwarty, a dane nie są nigdzie zapisywane
READ	plik tylko do odczytu	Nowy plik nie jest tworzony	Stary plik jest otwierany i można pobrać z niego dane, ale nie można go edytować
RECREATE	utworzenie na nowo	Utworzony zostaje nowy plik	Stary plik jest kasowany i zastępowany nowym
UPDATE	wprowadzenie nowych danych do pliku	Utworzony zostaje nowy plik	Nowe dane zostają dodane do już istniejących

Jeżeli przy definicji pliku nie zostanie wpisana opcja, to domyślnie działającą opcją jest "READ".

Opcje "CREATE" i "NEW" działają identycznie.

Histogramy

- **histogram jednowymiarowy**

> TH1F *hist = new TH1F("h1","Histogram",100,0.,10.)

"h1" - nazwa, "Histogram" - wyświetlany tytuł

100 - liczba binów, 0. - dolna krawędź, 10. - górna krawędź

- **histogram dwuwymiarowy**

> TH2F *hist2 = new TH2F("h2","Histogram",100,0.,10.,100,0.,10.);

- **rysowanie**

> hist->Draw();

Histogramy - wypełnianie

- **wypełnianie binów**

```
“for(Int_t i=0; i<1000; i++)  
{hist->Fill(1);}”
```

Jest to
wypełnianie
zdarzenie po
zdarzeniu.

bin o numerze 1 zostaje wypełniony
tysiącem zdarzeń

```
“TRandom3 r;  
r.SetSeed();  
for(Int_t i=0; i<1000; i++)  
{hist->Fill(r.Gaus(0.,1.));}”
```

Liczba
powtórzeń
pętli powinna
być równa
liczbie zdarzeń.

wypełnienie histogramu tysiącem
losowym zdarzeń, wg rozkładu Gaussa

- **ustalanie wartości binów**

```
“for(Int_t i=0; i<100; i++)  
{hist->SetBinContent(i,3);}”
```

Jest to wypełnianie
bin po binie, gdzie
wiemy już ile
zdarzeń będzie w
każdym z binów.

każdy bin będzie miał trzy zdarzenia

```
“for(Int_t i=0; i<100; i++)  
{hist->SetBinContent(i, tab[i] );}”
```

każdy bin będzie miał liczbę zdarzeń
zgodną z wcześniej przygotowaną
tablicą tab[i]

Liczba
powtórzeń
pętli
powinna być
równa liczbie
binów.

Wykresy

- **wykres**

TGraph plot (10,x,y);

10 - liczba rysowanych punktów, x[] - tablica z wartościami x, y[] - tablica z wartościami y

- **przykład**

```
"Double_t a[] = {1.,2.,3.};
```

```
Double_t b[]={0.,1.,0.};
```

```
TGraph plot(3,a,b);
```

```
plot.Draw();"
```

- **wykres pobierający dane z pliku**

```
"TGraph plot("dane.dat");"
```


TBrowser

okienkowa przeglądarka (wymagana włączona grafika)

umożliwia otwieranie plików .root i oglądanie zapisanych w nich histogramów oraz wykresów

możliwa jest edycja histogramów (zmiana opcji rysowanie, kolorów, zakresu, liczby binów itd.)

- **uruchamianie**

> root

> new TBrowser

TBrowser

