sPHENIX INTT Analysis Workshop,
November 20th, 2024
Korea University, Republic of Korea

Annyeonghaseyo

안녕하세요 Fun4All!
Fun4All Tutorial for beginners

G. Nukazuka (RIKEN/RBRC)

Table of contents

Hands on list

About this talk

This talk presents how to run your analysis codes in the Fun4All framework. Audiences are asked to download/run/change some codes, so you need to have a BNL account.

Hands-on Program

1. Downloading the sample codes
2. Checking your environmental variables
3. Running the minimal code (Fun4All_minimum.C)
4. Making/compiling your analysis module
5. Modifying your environmental variables to include your analysis module
6. Modifying and running the sample codes (Fun4All_minimum_2.C, 3, and 4)

what/who
why
where
when→now
how

What is Fun4All?

An analysis framework originally developed for the PHENIX experiment

what/who
why
where
when→now
how

What is Fun4All?

An analysis framework originally developed for the PHENIX experiment

Software framework

37 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

"Framework (computer science)" redirects here. For other uses, see Framework (disambiguation).

In computer programming, a **software framework** is an abstraction in which software, providing generic functionality, can be selectively changed by additional user-written code, thus providing application-specific software. It provides a standard way to build and deploy applications and is a universal, reusable software environment that provides particular functionality as part of a larger software platform to facilitate the development of software applications, products and solutions.

軟體框架

37种语言

条目 讨论 汉 漢 不转换 阅读 编辑 查看历史 工具

维基百科，自由的百科全书

此條目**沒有列出任何参考或来源**。 *(2016年8月3日)*
維基百科所有的內容都應該可供查證。请协助補充可靠来源以改善这篇条目。无法查证的內容可能會因為異議提出而被移除。

軟體框架（software framework），通常指的是為了實現某個業界標準或完成特定基本任務的軟體組件規範，也指為了實現某個軟體組件規範時，提供規範所要求之基礎功能的軟體產品。

框架的功能類似於基礎設施，與具體的軟體應用無關，但是提供並實現最為基礎的軟體架構和體系。軟體開發者通常依據特定的框架實現更為複雜的商業運用和業務邏輯。這樣的軟體應用可以在支持同一種框架的軟體系統中運行。

簡而言之，框架就是制定一套規範或者規則（思想），大家（程序员）在該規範或者規則（思想）下工作。或者說使用別人搭好的舞台來做編劇和表演。

소프트웨어 프레임워크

37개 언어

문서 토론 읽기 편집 역사 보기 도구

위키백과, 우리 모두의 백과사전.

컴퓨터 프로그래밍에서 **소프트웨어 프레임워크**(software framework)는 복잡한 문제를 해결하거나 서술하는 데 사용되는 기본 개념 구조이다. 간단히 **뼈대**, **골조**(骨組), **프레임워크**(framework)라고도 한다. 이렇게 매우 폭넓은 정의는 이 용어를 버즈워드(buzzword)로서, 특히 소프트웨어 환경에서 사용할 수 있게 만들어 준다.

ソフトウェアフレームワーク

37の言語版

ページ ノート 閲覧 編集 履歴表示 ツール

出典: フリー百科事典『ウィキペディア（Wikipedia）』

ソフトウェアフレームワーク（英: software framework）とは、プログラミングにおいて、アプリケーションソフトウェア等の実装に必要となる一般的な機能や定型コードを、ライブラリとしてあらかじめ用意したものである。例えば、Javaなどのオブジェクト指向言語向けのクラスライブラリとして実装されている場合は、再利用可能なソフトウェア部品（ソフトウェアコンポーネント）として用意されているクラスのインスタンスを自由に組み合わせたり、基本的な機能を持つ基底クラスを継承した派生クラスをユーザープログラマーが定義し、仮想メソッドによって公開されているカスタマイズポイントを選択的に上書きしたり特化させたりする。言語によってはコールバック関数やデリゲートを利用するなど、他にもさまざまな形態がある。文脈から明確な場合は単に「フレームワーク」としたり、特にアプリケーションソフトウェア開発向けであることを明確にした「アプリケーションフレームワーク」など、前後に別の語をつなげた複合語を使ったりすることもある。

READ LATER

Wikipedia

what/who
why
where
when→now
how

What is Fun4All?

An analysis framework originally developed for the PHENIX experiment

what/who
why
where
when→now
how

Why do we use Fun4All?

- Fun4All has a successful history.
- Fun4All has useful features.
- Other sPHENIX members use it.
- **Only analysis results done with Fun4All can be published from sPHENIX.**

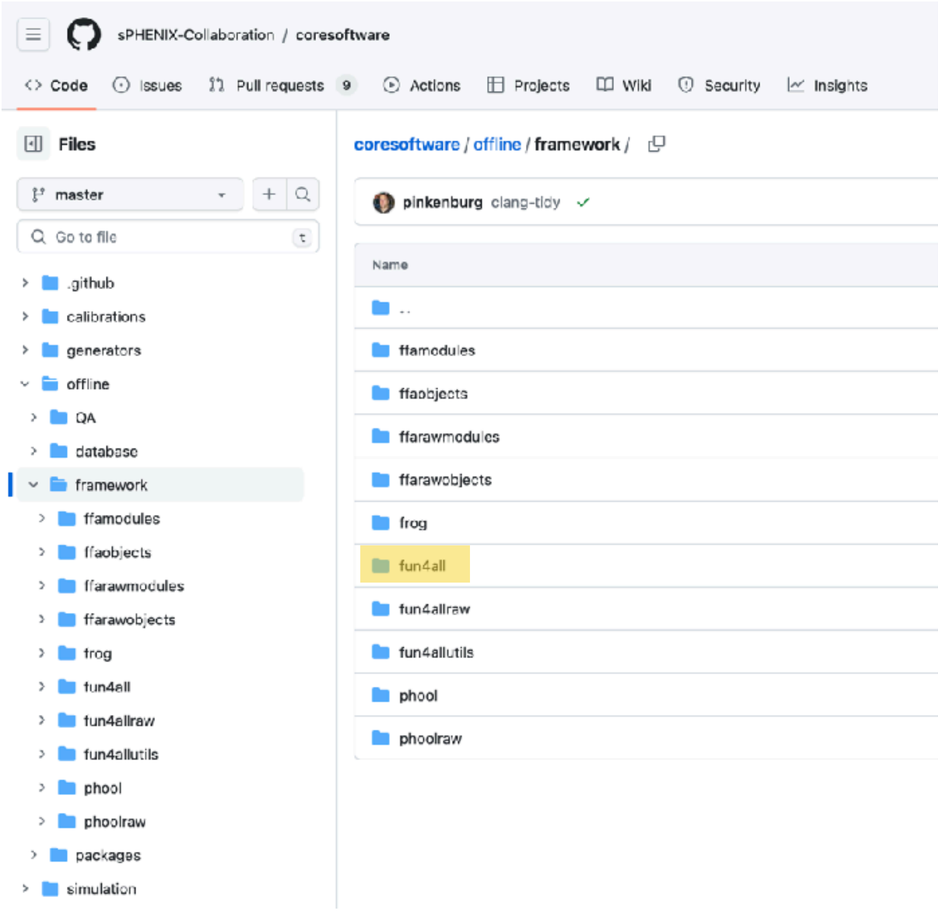
what/who
why
where
when→now
how

Where is Fun4All?

- You can find it on GitHub: https://github.com/sPHENIX-Collaboration/coresoftware
- You can use it in the SDCC servers.

Steps to set up Fun4All in SDCC

1. Log in to the SDCC gateway machine:
```
$ ssh {username}@ssh.sdcc.bnl.gov
```
2. Log in to the SDCC servers:
```
$ ssh {username}@sphnx{num}sdcc.bnl.gov
```
{num}: 01 — 08
3. Execute the setting shell script:
```
$ source /opt/sphenix/core/bin/sphenix_setup.sh
```



How?

This is the question!

Why is Fun4All difficult?

Why is Fun4All difficult?

Git
GitHub

C++

automake
configure

ROOT
macro

Linux

You need

a variety of knowledge

and experience

Compiling
make

Fun4All

Static
library

Finding
information

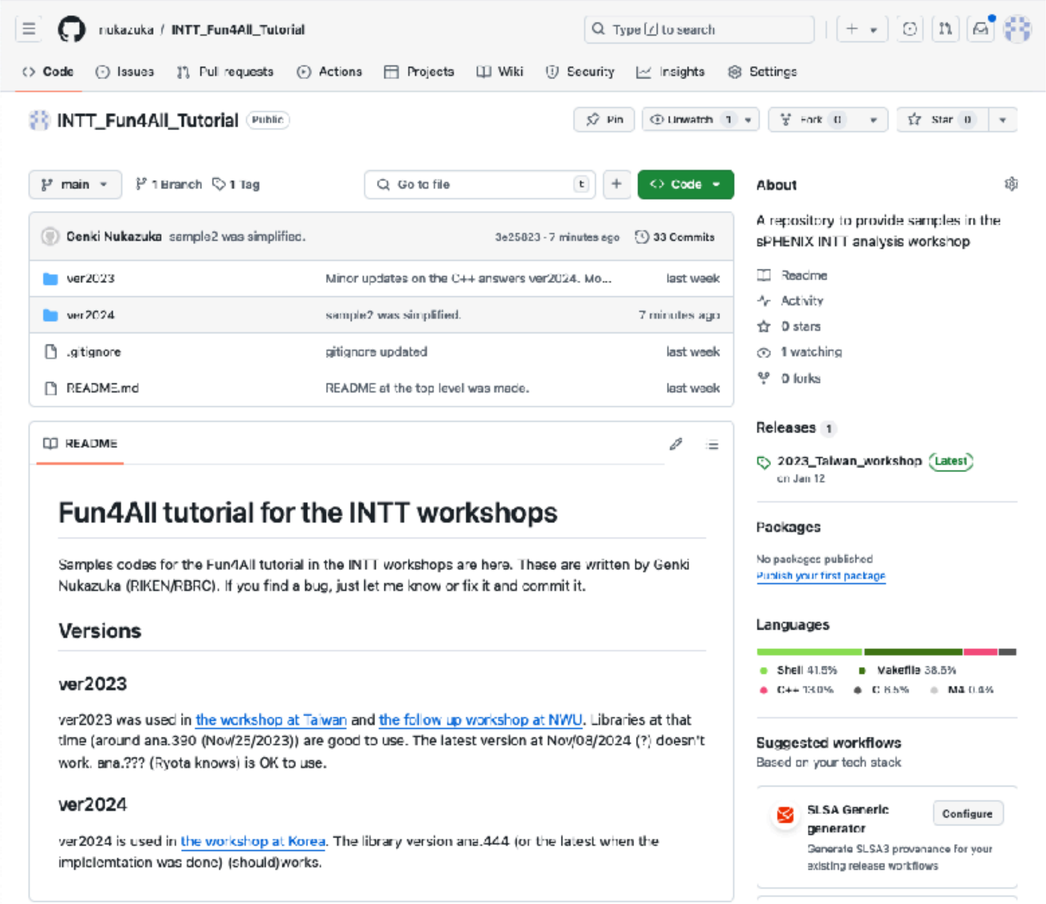
sPHENIX
scripts

Are you a Fun4All beginner?

This tutorial is basically same as the one in the last year. It may (should) be boring for PhD students and 2nd grade graduate students as they already took it. **So for those good at Fun4All, please help others.**

INTT_Fun4All_Tutorial repository

https://github.com/nukazuka/INTT_Fun4All_Tutorial
It belongs to my private account but not sPHENIX.



https://github.com/nukazuka/INTT_Fun4All_Tutorial/tree/main

Get the sample codes

HANDS ON! #1

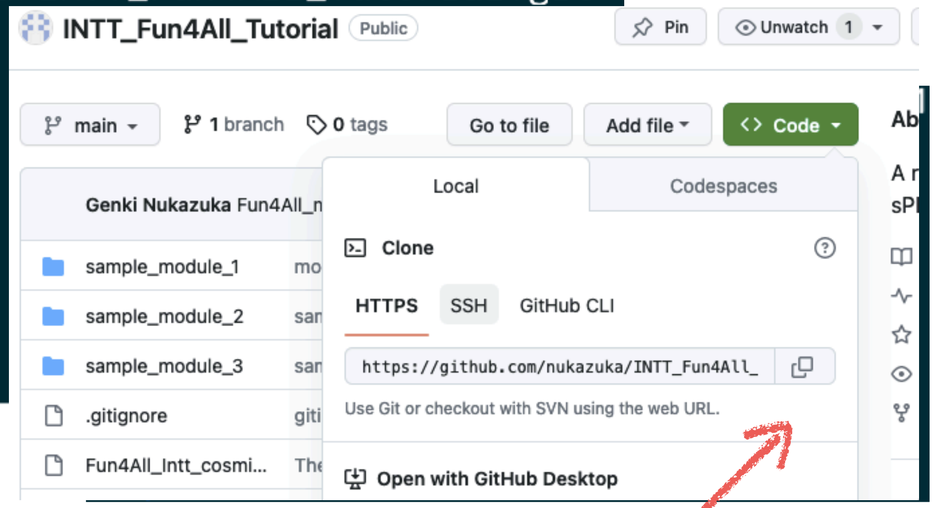
You can get the sample codes: https://github.com/nukazuka/INTT_Fun4All_Tutorial

1. Make a working directory under /sphenix/tg/tg01/commissioning/INTT/work/[yours] or anywhere you like.
2. Get them by

```
$ git clone git@github.com:nukazuka/INTT_Fun4All_Tutorial.git
```

in your working directory

```
[genki 17:55:14 fun4all_tutorial] $ git clone git@github.com:nukazuka/INTT_Fun4All_Tutorial.git
Cloning into 'INTT_Fun4All_Tutorial'...
X11 forwarding request failed on channel 0
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (33/33), done.
remote: Total 50 (delta 8), reused 47 (delta 8), pack-reused 0
Receiving objects: 100% (50/50), 443.36 KiB | 417.00 KiB/s, done.
Resolving deltas: 100% (8/8), done.
```



back to ToC

INTT_Fun4All_Tutorial repository

Structure:

```
[nukazuka@sphnx05 08:56:08 INTT_Fun4All_Tutorial] $ tre
.
├── ver2023
├── ver2024
│   ├── Fun4All_samples
│   │   ├── Fun4All_minimum.C
│   │   ├── Fun4All_minimum_2.C   Fun4All macro
│   │   ├── sample_module_2       directory for analysis module of this sample
│   │   │   ├── Makefile           Makefile
│   │   │   ├── Makefile.am        configuration file for autogen.sh
│   │   │   ├── autogen.sh         shell script to setup for compilation
│   │   │   ├── configure          configure
│   │   │   ├── configure.ac       configure.ac (configuration file for autogen.sh)
│   │   │   ├── tutorial.cc        some more files for the compilation
│   │   │   ├── tutorial.h         source file of analysis module
│   │   │   └── (etc…)             header file of analysis module
│   │   ├── sample_module_2
│   │   ├── Fun4All_minimum_3.C
│   │   ├── sample_module_3
│   │   ├── Fun4All_minimum_4.C
│   │   ├── sample_module_4
│   │   ├── Fun4All_minimum_5.C
│   │   ├── sample_module_5
│   │   ├── Fun4All_minimum_6.C
│   │   └── sample_module_6
│   └── cpp_basics
└── README.md
```

https://github.com/nukazuka/INTT_Fun4All_Tutorial

It belongs to my private account but not sPHENIX.

INTT_Fun4All_Tutorial repository

- sample1
  - Exercise for running Fun4All.
  - It contains minimum codes to show the simplest case. So it does not require any analysis module and does nothing.
  - Also, we will check the configuration of your environment. If you have no idea what to do, I'll show an example.
- sample2
  - You can see how to add your analysis module.
  - The analysis module (sample_module_2) just prints words on your terminal.
- sample3
  - InttRawHit is taken from a given DST and analyzed using the sample analysis module (sample_module_3).
- sample4
  - TrkrCluster is taken from a given DST and analyzed using the sample analysis module (sample_module_4).

Different from last year

Different from last year

INTT_Fun4All_Tutorial repository: Read README.md

INTT_Fun4All_Tutorial Public

Pin | Unwatch 1

main | 1 Branch | 1 Tag | Go to file | Add file | Code

Genki Nukazuka sample2 was simplified. 3e26823 · 17 hours ago 33 Commits

ver2023	Minor updates on the C++ answers ver2024. More explan...	last week
ver2024	sample2 was simplified.	17 hours ago
.gitignore	gitignore updated	last week
README.md	README at the top level was made.	last week

README

Fun4All tutorial for the INTT workshops

Samples codes for the Fun4All tutorial in the INTT workshops are here. These are written by Genki Nukazuka (RIKEN/RBRC). If you find a bug, just let me know or fix it and commit it.

Versions

ver2023

ver2023 was used in the workshop at Taiwan and the follow up workshop at NWU. Libraries at that time (around ana.390 (Nov/25/2023)) are good to use. The latest version at Nov/08/2024 (?) doesn't work. ana.??? (Ryota knows) is OK to use.

ver2024

ver2024 is used in the workshop at Korea. The library version ana.444 (or the latest when the implelemtation was done) (should)works.

Name	Last commit message	Last commit date
..		
Fun4All_samples	sample2 was simplified.	17 hours ago
cpp_basics	Minor updates on the C++ answers ver2024. More ex...	last week
README.md	README.md updated	now

README.md

INTT_Fun4All_Tutorial in 2024 at Korea Univ.

A repository to provide samples in the sPHENIX INTT analysis workshops in Korea. Files under cpp_basics were used in the workshop at NWU in Jan/2024.

README.md

Samples for Fun4All tutorial

Tutorial samples

Step1

The minimum sample. It's just a practice to run Fun4All.

Files

- Fun4All_minimum.C

Step2

It shows how to add your own analysis modu

Files/Directories

- Fun4All_minimum_2.C
- sample_module_2

Step3

See INTT hits from MC events.

Files/Directories

- Fun4All_minimum_3.C
- sample_module3

Step4

A DST file is read, and information of INTTR

Preview | Code | Blame | Raw

Sample 2

This analysis module just prints words on your terminal like:

```
[nukazuka@sphnx05 09:49:10 Fun4All_samples] $ root -q -b Fun4All_minimum_2.C ; echo $?

Processing Fun4All_minimum_2.C...
tutorial::tutorial(const std::string &name) Calling ctor
tutorial::Init(PHCompositeNode *topNode) Initializing
Fun4AllServer::setRun(): run 0 uses CDB TIMESTAMP 0
tutorial::InitRun(PHCompositeNode *topNode) Initializing for Run XXX
-------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      RUN (PHCompositeNode)/
         PAR (PHCompositeNode)/


tutorial::process_event(PHCompositeNode *topNode) Processing Event
tutorial::ResetEvent(PHCompositeNode *topNode) Resetting internal structures, prepare for
tutorial::EndRun(const int runnumber) Ending Run for Run 0
tutorial::End(PHCompositeNode *topNode) This is the End...
tutorial::Reset(PHCompositeNode *topNode) being Reset
tutorial::~tutorial() Calling dtor
0[nukazuka@sphnx05 09:49:10 Fun4All_samples] $ root -q -b Fun4All_minimum_2.C ; echo $?
```

README is a commonly used text file to tell what's inside. I put README.md in each directory for the explanation for the directory. It can be your hint. Read it!

sample 1

What can we begin with?

A minimum program is good to start with.

In the case of C++:

sample 1

What can we begin with?

A minimum program is good to start with.

In the case of C++:

```
[genki 18:29:48 fun4all_tutorial] $ /bin/cat cpp_minimum.cc
int main(){}
[genki 18:29:49 fun4all_tutorial] $ g++ cpp_minimum.cc
[genki 18:29:56 fun4all_tutorial] $ ./a.out
```

It's useless, I know.

In the case of a ROOT macro:

```
[genki 18:31:41 fun4all_tutorial] $ /bin/cat root_minimum.cc
void root_minimum(){}
[genki 18:31:45 fun4all_tutorial] $ root root_minimum.cc
root [0]
Processing root_minimum.cc...
root [1] .q
```

It's also useless.

sample 1

What can we begin with?

In the case of Fun4All:

```c
#include <fun4all/Fun4AllServer.h>

R__LOAD_LIBRARY(libfun4all.so)

int Fun4All_minimum()
{

  Fun4AllServer *se = Fun4AllServer::instance();

  se->run( 1 );
  se->End();
  delete se;

  gSystem->Exit(0);
  return 0;
}
```

Fun4All_minimum.C

This is a ROOT macro.

terminal

```
[nukazuka@sphnx04 22:38:47 tutorial] $  root -q -b Fun4All_minimum.C

Processing Fun4All_minimum.C...
Fun4AllServer::setRun(): could not get timestamp for run  0, using t
ics(0) timestamp: Wed Dec 31 19:00:00 1969
-------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
   RUN (PHCompositeNode)/
   PAR (PHCompositeNode)/
```

Let’s see the sample code line by line.

sample 1

What can we start with?

```
1 #include <fun4all/Fun4AllServer.h>
```

include statement to include fun4all/Fun4Allserver.h
To find the file

You need to execute a shell script provided by sPHENIX to set up analysis environment:

```
$ source /opt/sphenix/core/bin/sphenix_setup.sh
```

0. ROOT_INCLUDE_PATH is one of the environmental variables. ROOT uses it to find files to be included. Let's see all environmental variables:

```
$ env
```

```
[nukazuka@sphnx05 12:28:04 Fun4All_samples] $ env
G4LEVELGAMMADATA=/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/geant4.
MANPATH=/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c96
355ed/x86_64-centos7/share/man:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/re
vmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/utils/man:/cvmfs/sphenix.sdcc.b
core/root-6.26.06.p01/man:/usr/local/share/man:/usr/share/man
XDG_SESSION_ID=809180
HOSTNAME=sphnx05.sdcc.bnl.gov
ROOT_INCLUDE_PATH=/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/macros/de
/Fun4All_codes/install/include:/sphenix/tg/tg01/commissioning/INTT/work/genki/re
ommissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/
_preliminary/202409_performance/event_display/install/include/inttread:/sphenix/
henix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmi
eneral_codes/Jaein/Cosmics/install/include/analysisinttcosmiccommissioning:/sphe
/inttcosmicsbcofinder:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/cores
oresoftware/calibrations/intt/install/include/inttcalib:/sphenix/tg/tg01/commiss
tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/offline/packages/insta
nstall/include/intt:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresof
resoftware2/calibrations/intt/install/include/inttcalib:/sphenix/tg/tg01/commiss
g01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/in
t/offline/framework/install/include/mvtx_decoder:/sphenix/tg/tg01/commissioning/
```

sample 1

Environmental variables: Genki's case

```
$ env | tr = " " | awk '{print $1}'
```

- G4LEVELGAMMADATA
- MANPATH
- XDG_SESSION_ID
- HOSTNAME
- **ROOT_INCLUDE_PATH**
- OPT_UTILS
- TERM
- SHELL
- EVT_LIB
- HISTSIZE
- LHAPATH
- ORIG_LD_LIBRARY_PATH
- SSH_CLIENT
- PERL5LIB
- LHAPDF_DATA_PATH
- G4_MAIN
- QTDIR
- OLDPWD
- QTINC
- SSH_TTY
- G4LEDATA
- QT_GRAPHICSSYSTEM_CHECKED
- USER
- LD_LIBRARY_PATH
- LS_COLORS
- G4NEUTRONHPDATA
- PGUSER
- SSH_AUTH_SOCK
- G4ENSDFSTATEDATA
- G4RADIOACTIVEDATA
- CONFIG_SITE
- G4ABLADATA
- MAIL
- PATH
- OPT_SPHENIX
- PYTHIA8
- G4PIIDATA
- PWD
- G4PARTICLEXSDATA
- LANG
- NOPAYLOADCLIENT_CONF
- PGHOST
- MODULEPATH
- GSEARCHPATH
- PARASOFT
- QT_GRAPHICSSYSTEM
- LOADEDMODULES
- KDEDIRS
- OFFLINE_MAIN
- ITERM_ORIG_PS1
- PS1
- XPLOAD_CONFIG_DIR
- G4SAIDXSDATA
- CXX
- XERCESCROOT
- ROOTSYS
- HISTCONTROL
- CALIBRATIONROOT
- SHLVL
- HOME
- G4REALSURFACEDATA
- ORIG_MANPATH
- ITERM_PREV_PS1
- FC
- PYTHONPATH
- ORIG_PATH
- DCACHE_RA_BUFFER
- LOGNAME
- QTLIB
- CVS_RSH
- SSH_CONNECTION
- MODULESHOME
- COMPILER_PATH
- LESSOPEN
- OPT_FUN4ALL
- CC
- XDG_RUNTIME_DIR
- DISPLAY
- ONLINE_MAIN
- QT_PLUGIN_PATH
- G4INCLDATA
- NO_AT_BRIDGE
- INTT_WORK

sample 1

What can we start with?

```
1 #include <fun4all/Fun4AllServer.h>
```

include statement to include fun4all/Fun4Allserver.h

To find the file

1. Check the environment variable ROOT_INCLUDE_PATH:

```
$ echo $ROOT_INCLUDE_PATH
```

```
[nukazuka@sphnx04 22:38:53 tutorial] $ echo $ROOT_INCLUDE_PATH
./:/sphenix/tg/tg01/commissioning/INTT/repositories/tutorials/AnaTutorial/install/include:/sphenix/tg/tg01/commissioning/INTT/repositories/tutorials/AnaTutorial/install/include
/anatutorial:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/hachiya/F4AInttRead/install/include:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/IN
TT/general_codes/hachiya/F4AInttRead/install/include/inttread:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/include:/sphen
ix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/include/inttanalysiscosmic:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/co
resoftware/simulation/g4simulation/g4intt/install/include:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/simulation/g4simulation/g4intt/install/include/g4int
t:/sphenix/tg/tg01/commissioning/INTT/repositories/libraries/include:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include:/cvmfs/sphenix.sdcc.bnl.gov/gcc-
12.1.0/release/release_ana/ana.382/include:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/ffarawobjects:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/relea
se/release_ana/ana.382/include/JSON:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/half:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/a
na.382/include/torch:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g4detectors:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/i
nclude/eventplane:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/kineto:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g
4decayer:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/phfield:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/LHAPDF:/c
vmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/c10:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/oneapi:/cvmfs/sphenix.sd
cc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/DDCond:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g4tracking:/cvmfs/sphenix.sdcc.bnl.g
ov/gcc-12.1.0/release/release_ana/ana.382/include/litecaloeval:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g4intt:/cvmfs/sphenix.sdcc.bnl.gov/gcc
-12.1.0/release/release_ana/ana.382/include/phool:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/boost:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/releas
e/release_ana/ana.382/include/Pythia8Plugins:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/calib_emc_pi0:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/rel
ease/release_ana/ana.382/include/ffaobjects:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/EvtGenBase:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release
/release_ana/ana.382/include/flowafterburner:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/google:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/re
```

It’s not human-readable. Paths are separated by “:”. Let’s make it better.

You need to execute a shell script provided by sPHENIX to set up analysis environment:

```
$ source /opt/sphenix/core/bin/sphenix_setup.sh
```

sample 1

What can we start with?

```
1 #include <fun4all/Fun4AllServer.h>
```

include statement to include fun4all/Fun4Allserver.h

To find the file

1. Check the environment variable ROOT_INCLUDE_PATH:

```
$ echo $ROOT_INCLUDE_PATH
```

2. To separate the paths: Log in to the SDCC servers:

```
$ echo $ROOT_INCLUDE_PATH | tr : "\n"
```

UPDATED

tr command replaces : to \n.

```
[nukazuka@sphnx04 22:48:37 tutorial] $ sed_path $ROOT_INCLUDE_PATH
./
/sphenix/tg/tg01/commissioning/INTT/repositories/tutorials/AnaTutorial/install/include
/sphenix/tg/tg01/commissioning/INTT/repositories/tutorials/AnaTutorial/install/include/anatutorial
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/hachiya/F4AInttRead/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/hachiya/F4AInttRead/install/include/inttread
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/include/inttanalysiscosmic
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/simulation/g4simulation/g4intt/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/simulation/g4simulation/g4intt/install/include/g4intt
/sphenix/tg/tg01/commissioning/INTT/repositories/libraries/include
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/ffarawobjects
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/JSON
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/half
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/torch
```

Much better! Let's find paths which have a certain word.

sample 1

READ LATER

tr command

```
- Replace all occurrences of a character in a file, and print the result:
    tr find_character replace_character < path/to/file

- Replace all occurrences of a character from another command's output:
    echo text | tr find_character replace_character

- Map each character of the first set to the corresponding character of the second set:
    tr 'abcd' 'jkmn' < path/to/file

- Delete all occurrences of the specified set of characters from the input:
    tr -d 'input_characters' < path/to/file

- Compress a series of identical characters to a single character:
    tr -s 'input_characters' < path/to/file

- Translate the contents of a file to upper-case:
    tr "[:lower:]" "[:upper:]" < path/to/file

- Strip out non-printable characters from a file:
    tr -cd "[:print:]" < path/to/file
```

sample 1

What can we start with?

```
1 #include <fun4all/Fun4AllServer.h>
```

include statement to include fun4all/Fun4Allserver.h

To find the file

1. Check the environment variable ROOT_INCLUDE_PATH:
   ```
   $ echo $ROOT_INCLUDE_PATH
   ```
2. To separate the paths: Log in to the SDCC servers:
   ```
   $ echo $ROOT_INCLUDE_PATH | tr : "\n"
   ```
   tr command replaces : to \n.
3. Select paths which contain fun4all
   ```
   $ echo $ROOT_INCLUDE_PATH | tr : "\n" | grep fun4all
   ```

```
[nukazuka@sphnx05 12:36:09 Fun4All_samples] $ sed_path $ROOT_INCLUDE_PATH | grep fun4all
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/include/fun4allraw
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4all        ← this one!
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4allraw
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4allutils
```

sample 1

What can we start with?

```
1 #include <fun4all/Fun4AllServer.h>
```

include statement to include fun4all/Fun4Allserver.h

To find the file

1. Check the environment variable ROOT_INCLUDE_PATH:
   ```
   $ echo $ROOT_INCLUDE_PATH
   ```
2. To separate the paths: Log in to the SDCC servers:
   ```
   $ echo $ROOT_INCLUDE_PATH | tr : "\n"
   ```
   tr command replaces : to \n.
3. Select paths which contain fun4all
   ```
   $ echo $ROOT_INCLUDE_PATH | tr : "\n" | grep fun4all
   ```
4. Confirmation
   ```
   $ ls /cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4all/Fun4AllServer.h
   ```

HANDS ON! #2

Try them

replace it with your case

```
[nukazuka@sphnx05 12:36:09 Fun4All_samples] $ sed_path $ROOT_INCLUDE_PATH | grep fun4all
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/include/fun4allraw
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4all        ← this one!
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4allraw
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include/fun4allutils
```

sample 1

What can we start with?

Rtypes.h

◆ R__LOAD_LIBRARY

```
#define R__LOAD_LIBRARY ( LIBRARY )
```

Definition at line 467 of file Rtypes.h.

```
#include <fun4all/Fun4AllServer.h>

R__LOAD_LIBRARY(libfun4all.so)
```

Learn C language more if you don't know.

R__LOAD_LIBRARY is a function-like macro defined in ROOT to load a library.
A shared library libfun4all.so is loaded. Where is it?

1. Check the environment variable LD_LIBRARY_PATH:

```
$ echo $LD_LIBRARY_PATH
```

```
[nukazuka@sphnx04 23:12:14 tutorial] $ echo $LD_LIBRARY_PATH
/sphenix/tg/tg01/commissioning/INTT/repositories/tutorials/AnaTutorial/install/lib:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/hachiya/F4AInttRead/i
nstall/lib:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/lib:/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoft
ware/simulation/g4simulation/g4intt/install/lib:/sphenix/tg/tg01/commissioning/INTT/repositories/libraries/lib:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.
0-57c96/x86_64-centos7/lib:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c96/x86_64-centos7/lib64:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core
/binutils/2.37-355ed/x86_64-centos7/lib:.:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/lib64:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/an
a.382/lib:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/utils/lib64:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/utils/lib:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt
/sphenix/core/lib:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/root-6.26.06.p01/lib:/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/geant4.10.07.p04/lib64:/c
vmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/lhapdf-5.9.1/lib:/afs/rhic.bnl.gov/app/insure-7.5.5/lib:/usr/local/lib64:/usr/lib64
```

It's not human-readable again. Let's do the same.

sample 1

What can we start with?

Rtypes.h

◆ R__LOAD_LIBRARY

#define R__LOAD_LIBRARY (LIBRARY)

Definition at line 467 of file Rtypes.h.

```
#include <fun4all/Fun4AllServer.h>

R__LOAD_LIBRARY(libfun4all.so)
```

R__LOAD_LIBRARY is a function-like macro defined in ROOT to load a library.
A shared library libfun4all.so is loaded. Where is it?

1. Check the environment variable LD_LIBRARY_PATH:
   ```
   $ echo $LD_LIBRARY_PATH
   ```
2. Replce : to \n (or something else you like)
   ```
   $ echo $LD_LIBRARY_PATH | tr : "\n"
   ```

```
[nukazuka@sphnx05 12:42:23 Fun4All_samples] $ echo $LD_LIBRARY_PATH | tr : "\n"
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/event_display
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmics/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/calibrations/intt/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/offline/packages/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/calibrations/intt/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/correlation/i
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/timing/instal
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/QAhtml_repo/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/QA_codes/install/lib
```

It's better but still not clear… Let's search the file.

sample 1

What can we start with?

```
#include <fun4all/Fun4AllServer.h>

R__LOAD_LIBRARY(libfun4all.so)
```

R__LOAD_LIBRARY is a function-like macro defined in ROOT to load a library.
A shared library libfun4all.so is loaded. Where is it?

1. Check the environment variable LD_LIBRARY_PATH:
   ```
   $ echo $LD_LIBRARY_PATH
   ```
2. Replce : to \n (or something else you like):
   ```
   $ echo $LD_LIBRARY_PATH | tr : "\n"
   ```
3. Search libfun4all.so:
   ```
   $ echo $LD_LIBRARY_PATH | tr : "\n" | xargs -I {} find {} -name "libfun4all.so"
   ```

```
[nukazuka@sphnx05 12:43:58 Fun4All_samples] $ echo $LD_LIBRARY_PATH | tr : "\n" | xargs -I {} find {} -name "libfun4all.so"
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/lib/libfun4all.so
```

sample 1

What can we start with?

HANDS ON! #3

```
#include <fun4all/Fun4AllServer.h>

R__LOAD_LIBRARY(libfun4all.so)
```

R__LOAD_LIBRARY is a function-like macro defined in ROOT to load a library.
A shared library libfun4all.so is loaded. Where is it?

Try them

1. Check the environment variable LD_LIBRARY_PATH:
```
$ echo $LD_LIBRARY_PATH
```
2. Replce : to \n (or something else you like):
```
$ echo $LD_LIBRARY_PATH | tr : "\n"
```
3. Search libfun4all.so:
```
$ echo $LD_LIBRARY_PATH | tr : "\n" | xargs -I {} find {} -name "libfun4all.so"
```

Another way I could come up:
```
$ for dir in `echo $LD_LIBRARY_PATH | tr : "\n" ` ; do find $dir -name "libfun4all.so" ; done
```

back to ToC

sample 1

What can we start with?

```
1 #include <fun4all/Fun4AllServer.h>
2
3 R__LOAD_LIBRARY(libfun4all.so)
4
5 int Fun4All_minimum()
6 {
7
8   Fun4AllServer *se = Fun4AllServer::instance();
```

Including the header file and loading the shared library are for here.

A pointer of an instance of the Fun4AllServer class is assigned to “se”.

sample 1

What can we start with?

```
1  #include <fun4all/Fun4AllServer.h>
2
3  R__LOAD_LIBRARY(libfun4all.so)
4
5  int Fun4All_minimum()
6  {
7
8    Fun4AllServer *se = Fun4AllServer::instance();
9
10   se->run( 1 );
11   se->End();
12   delete se;
13
14   gSystem->Exit(0);
15   return 0;
16 }
```

Including the header file and loading the shared library are for here.

← Running analysis processes for the given number of events.

← Some processes are launched at the end of event-by-event processes.

← Just delete it.

← Just do it.

← Just do it.

This super simple macro takes no input file and outputs nothing. 1 event is processed.

sample 1

What can we start with?

```
[nukazuka@sphnx05 12:48:06 Fun4All_samples] $ root -q -b Fun4All_minimum.C

Processing Fun4All_minimum.C...
Fun4AllServer::setRun(): run 0 uses CDB TIMESTAMP 0
-----------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
   RUN (PHCompositeNode)/
   PAR (PHCompositeNode)/
```

HANDS ON! #4

Execute Fun4All_minimum.C.

Practical example

It depends on what you want to do. For example:

- inputting raw file(s)
- inputting DST file(s)
- Monte-Carlo as an input

- running someone’s analysis codes
- running your analysis codes

- Outputting results to DST file(s)
- Outputting results to histograms/TTrees

Practical example

It depends on what you want to do. For example:

- inputting raw file(s)
- inputting DST file(s)
- Monte-Carlo as an input

- running someone's analysis codes
- running your analysis codes

- Outputting results to DST file(s)
- Outputting results to histograms/TTrees

Let's try a simple case.

sample 2

Sample2: Add an analysis module

```
#include <fun4all/Fun4AllServer.h>

R__LOAD_LIBRARY(libfun4all.so)

// It should be tutorial.h of sample_moudle_2
#include <tutorial.h>
R__LOAD_LIBRARY( libtutorial.so )

int Fun4All_minimum_2( int nEvents = 1 )
{

  Fun4AllServer *se = Fun4AllServer::instance();

  tutorial* analysis_module = new tutorial( "name" );
  se->registerSubsystem( analysis_module );

  se->run(nEvents);
  se->End();
  delete se;

  gSystem->Exit(0);
  return 0;
}
```

added

added

Fun4All_minimum_2.C

Updated since last year

sample 2

Analysis module

You need to write your analysis codes in a certain class.
The class is called “analysis module”. Analysis modules need to

- inherit the SubsysReco class (class inheritance)
- implement functions in the SubsysReco (polymorphism)
- be registered to Fun4AllServer by Fun4AllServer::registerSubsystem

sample 2

Analysis module

You need to write your analysis codes in a certain class.
The class is called “analysis module”. Analysis modules need to

- inherit the SubsysReco class (class inheritance)
- implement functions in the SubsysReco (polymorphism)
- be registered to Fun4AllServer by Fun4AllServer::registerSubsystem

READ LATER

You can learn class inheritance (継承, 継承, 상속)

and polymorphism (多态, ポリモーフィズム, 다형성)

in C++ textbooks. It’s not easy to understand them without taking time to learn.

sample 2

Analysis module

The standard way to implement the class, add it to the ROOT macro, and run it is

1. generating a template by CreateSubsysRecoModule.pl ↔ or using existing analysis module

   ```
   $ CreateSubsysRecoModule.pl [name_of_the_module] [options]
   ```

   Joseph's minimum example is also a good start.
2. generating the configuration files by autogen.sh

   ```
   $ autogen.sh --prefix=[install_path]
   ```
3. implementing the header file (*.h) and the source file (*.cc) by yourself.
4. compiling the analysis module by make command

   ```
   $ make
   ```
5. installing the library (*.so) and the header file (*.h)

   ```
   $ make install
   ```
6. setting your LD_LIBRARY_PATH and ROOT_INCLUDE_PATH

   (here is a little bit complicated. The explanation is given later.)
7. adding an include statement and R__LOAD_LIBRARY macro to your ROOT macro, and execute it.

   (It's also given later.)

sample 2 SybsysReco class

Github

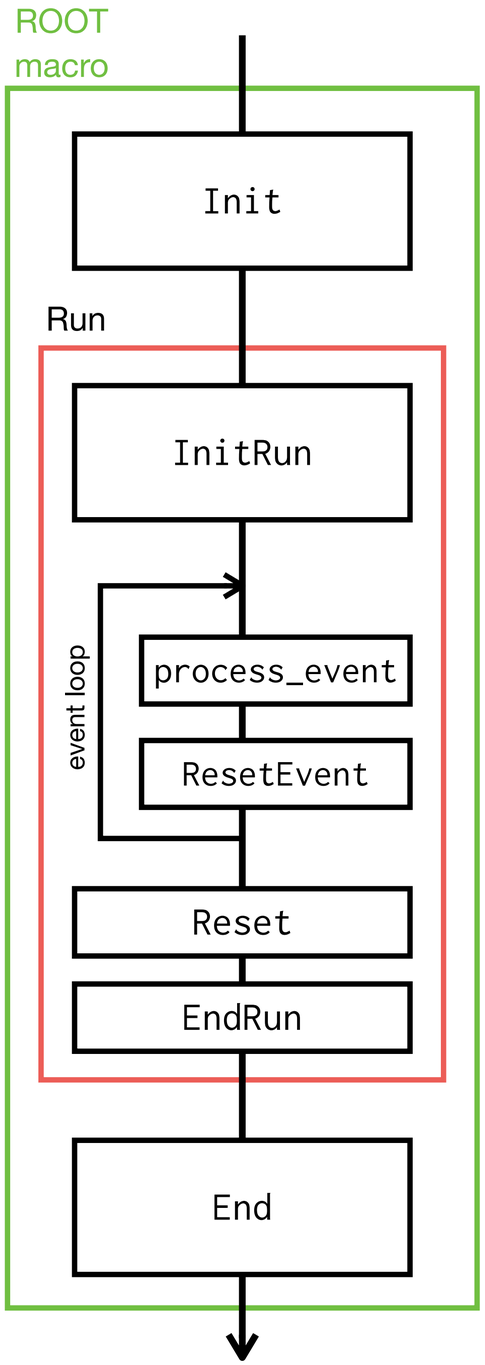
Code Blame 73 lines (58 loc) · 2.06 KB

```cpp
// Tell emacs that this is a C++ source
//  -*- C++ -*-.
#ifndef FUN4ALL_SUBSYSRECO_H
#define FUN4ALL_SUBSYSRECO_H

#include "Fun4AllBase.h"

#include <string>

class PHCompositeNode;

/** Base class for all reconstruction and analysis modules to be
 *  used under the Fun4All framework.
 *
 *  If you write a reconstruction/analysis module, you must derive
 *  from this base class and you have to implement this class methods.
 *  None of these are strictly required as far as C++ is concerned, but as
 *  far as your job is concerned, at least process_event(), to do the
 *  job, and InitRun(), to initialize, should be implemented.
 *
 */
```

```cpp
class SubsysReco : public Fun4AllBase
{
 public:
  /** dtor.
      Does nothing as this is a base class only.
  */
  ~SubsysReco() override {}

  /// Called at the end of all processing.
  virtual int End(PHCompositeNode * /*topNode*/) { return 0; }

  /// Called at the end of each run.
  virtual int EndRun(const int /*runnumber*/) { return 0; }

  /** Called during initialization.
      Typically this is where you can book histograms, and e.g.
      register them to Fun4AllServer (so they can be output to file
      using Fun4AllServer::dumpHistos() method).
   */
  virtual int Init(PHCompositeNode * /*topNode*/) { return 0; }

  /** Called for first event when run number is known.
      Typically this is where you may want to fetch data from
      database, because you know the run number.
   */
  virtual int InitRun(PHCompositeNode * /*topNode*/) { return 0; }

  /** Called for each event.
      This is where you do the real work.
  */
  virtual int process_event(PHCompositeNode * /*topNode*/) { return 0; }

  /// Reset.
  virtual int Reset(PHCompositeNode * /*topNode*/) { return 0; }

  /// Clean up after each event.
  virtual int ResetEvent(PHCompositeNode * /*topNode*/) { return 0; }

  void Print(const std::string & /*what*/ = "ALL") const override {}

 protected:
  /** ctor.
      @param name is the reference used inside the Fun4AllServer
  */
  SubsysReco(const std::string &name = "NONAME")
    : Fun4AllBase(name)
  {
  }
};

#endif
```

The only header is in Fun4All. The actual behavior of functions should be implemented in your inheriting class by yourself (polymorphism). The class itself is not too complicated.

back to ToC



SybsysReco/Your analysis module class

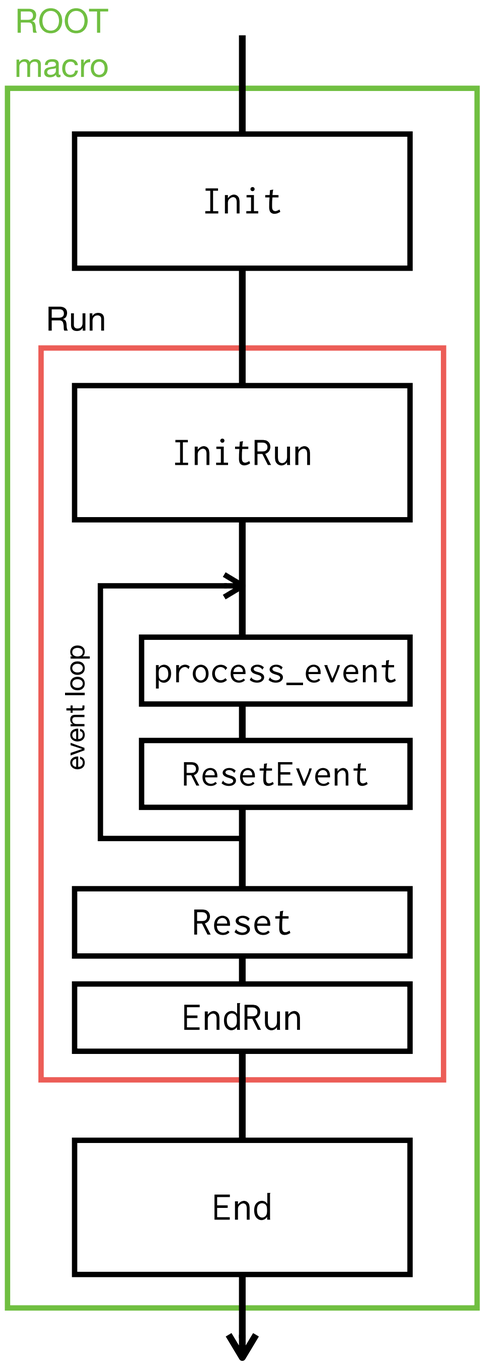
Github

The functions to be executed by Fun4AllServer take PHCompositNode* as an argument.
For example: `int process_event(PHCompositeNode *)`

sample 2

SybsysReco/Your analysis module class

Github



The functions to be executed by Fun4AllServer take PHCompositNode* as an argument.
For example: `int process_event(PHCompositeNode *)`

Preparation for the run. For example, making histograms.

The main part of your analysis

Finalization. Writing histogram objects to output files, etc.

sample 2

sample 2

Analysis module

The standard way to implement the class, add it to the ROOT macro, and run it is

1. generating a template by CreateSubsysRecoModule.pl

   ```
   $ CreateSubsysRecoModule.pl [name_of_the_module] [options]
   ```

   Joseph's minimum example is also a good start.

   Let's use sample_module_2

2. generating the configuration files by autogen.sh

   ```
   $ autogen.sh --prefix=[install_path]
   ```

3. implementing the header file (*.h) and the source file (*.cc) by yourself.
4. compiling the analysis module by make command

   ```
   $ make
   ```

5. installing the library (*.so) and the header file (*.h)

   ```
   $ make install
   ```

6. setting your LD_LIBRARY_PATH and ROOT_INCLUDE_PATH

   (here is a little bit complicated. The explanation is given later.)

7. adding an include statement and R__LOAD_LIBRARY macro to your ROOT macro, and execute it.

   (It's also given later.)

sample 2

make and Makefile

See the tutorial in the INTT workshop@NWU in Jan/2024

2024 / 01 / 17–18

sPHENIX INTT Analysis Workshop@NWU

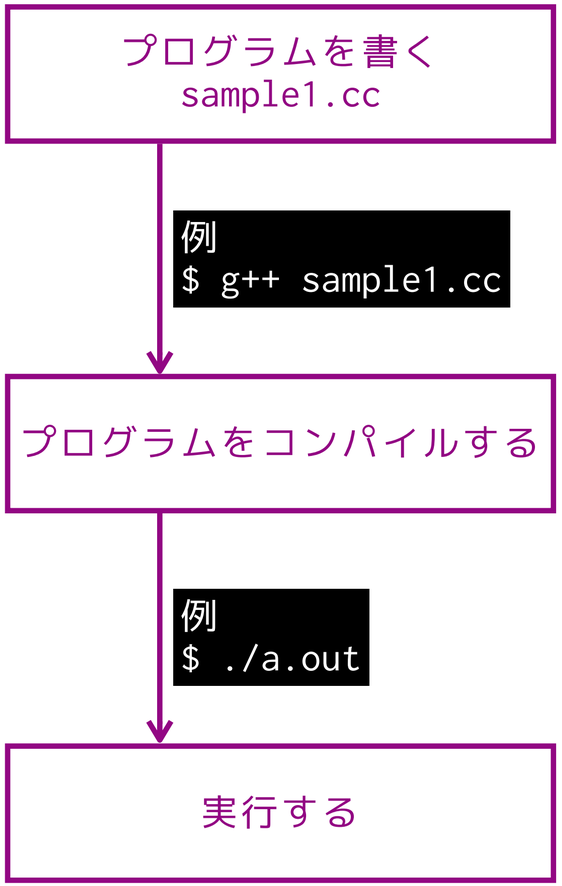
比較的大規模な プログラム構築のやり方

糠塚 元気 (理研/RBRC)

このチュートリアルでやること

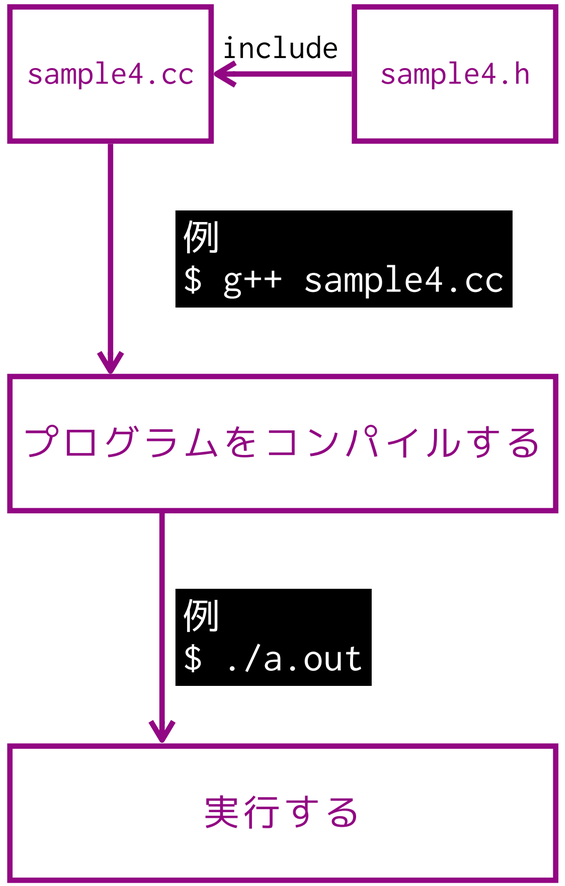
Sample1, 2, 3

準備運動



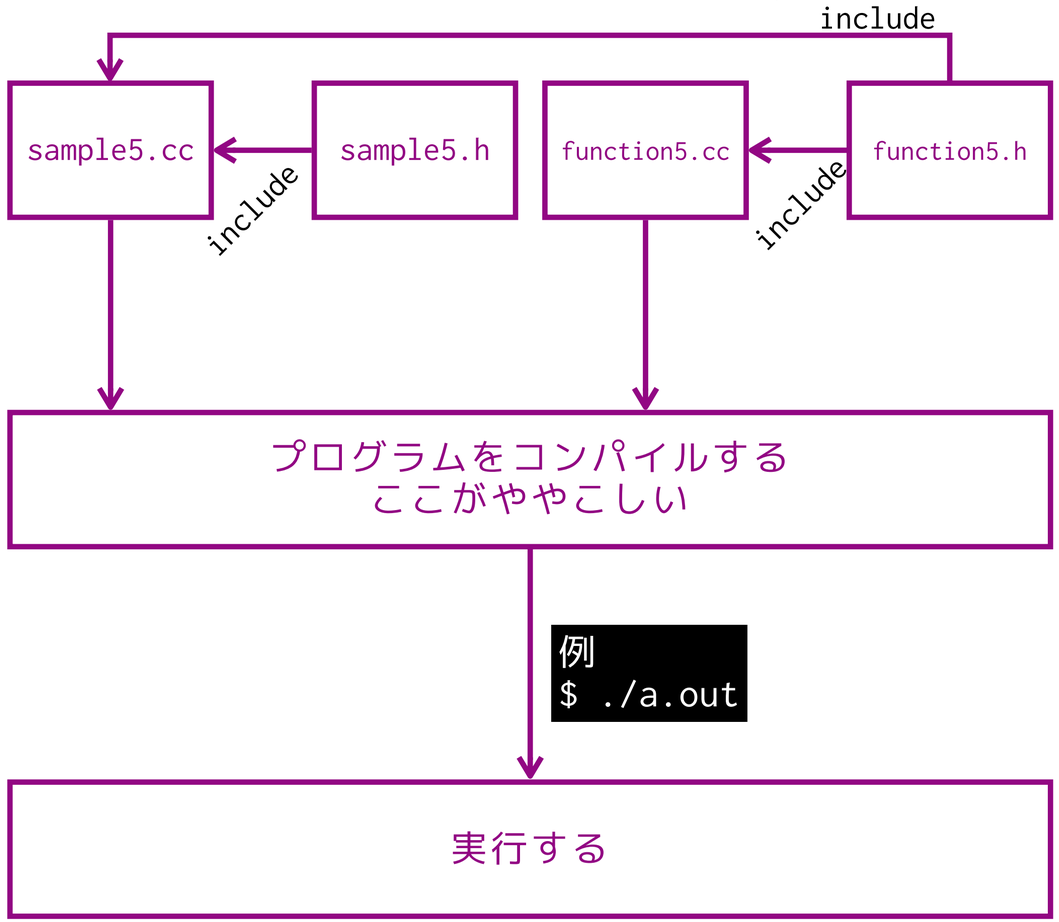
Sample4

メインコードの一部を
ヘッダーファイルに移す



Sample5

関数を別コードに移し、
メインコードと合わせてコンパイルする



C++ sample5: 関数をヘッダーファイルとソースファイルに分けて書く

sample1.cc sample2.cc sample3.cc function4.h **sample5.cc** Makefile

```cpp
/*
  A sample to use own function written in other header/source files.

  How To Compile
    1) Compile everything at once
      $ g++ sample5.cc function.cc

    2) Split compilation (分割コンパイル) by hand
      $ g++ -o sample5.o -c sample5.cc
      $ g++ -o function5.o -c function5.cc
      $ g++ sample5.o function5.o
      $ ./a.out

    3) Split compilation with make
      $ make sample5
      $ ./sample5
*/

#include <iostream>
#include "function5.h"

int main()
{
  double value = 2.0;
  std::cout << "Input value: " << value << std::endl;
  std::cout << "Twice of " << value << " is " << ReturnTwice( value ) << std::endl;

  return 0;
}
```

sample5.cc

```cpp
/* This also works as an include guard. */
#pragma once

// A declaration (宣言) of a function
double ReturnTwice( double value ) ;
```

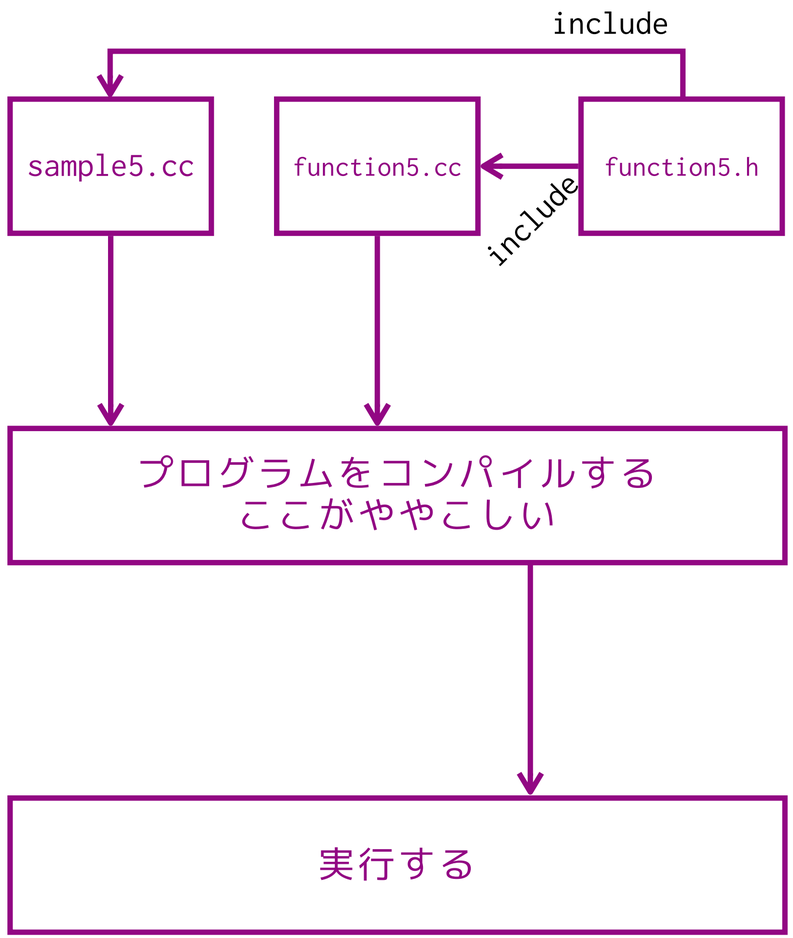
function5.h

U:--- **function5.h** All L1 (C/*l Abbrev)

```cpp
#include "function5.h"

// A definition (定義) of the function
double ReturnTwice( double value )
{
  return 2 * value;
}
```

function5.cc

C++ sample5: 関数をヘッダーファイルとソースファイルに分けて書く



サンプル 4 の function4.h の中身を

- double ReturnTwice(double x) の宣言を function5.h
- double ReturnTwice(double x) の定義を function5.h

に分けてみる

~20 分 実習

問題：複数のソースファイルをどう取り扱う？

A. 全部いっぺんにコンパイルする

```
$ g++ sample5.cc function5.cc
```

B. ソースファイルごとにコンパイルして、最後にまとめる

```
$ g++ -c sample5.cc
$ g++ -c function5.cc
$ g++ sample5.o function5.o
```

← 中間ファイル sample5.o 生成
← 中間ファイル function5.o 生成
← sample5.o と function5.o から a.out を作成

C. function5 から（共有）ライブラリを作成し、sample5.cc コンパイル時にライブラリを読み込む。

```
$ g++ -shared -fPIC -o libfunction5.so function5.cc
$ g++ -L. -lfunction5 sample5.cc
```

C++ sample5: 資料

コンパイル方法 B: ソースファイルごとにコンパイルして、最後にまとめる
（分割コンパイル）

```
[nukazuka@sphnx05 01:43:06 answers] $ ls
function4.h  function5.cc  function5.h  Makefile  sample1.cc  sample2.cc  sample3.cc  sample4.cc  sample5.cc
[nukazuka@sphnx05 01:43:07 answers] $
[nukazuka@sphnx05 01:43:10 answers] $ g++ -c function5.cc
[nukazuka@sphnx05 01:43:16 answers] $ ls
function4.h  function5.cc  function5.h  function5.o  Makefile  sample1.cc  sample2.cc  sample3.cc  sample4.cc  sample5.cc
[nukazuka@sphnx05 01:43:21 answers] $
[nukazuka@sphnx05 01:43:22 answers] $ g++ -c sample5.cc
[nukazuka@sphnx05 01:43:29 answers] $ ls
function4.h   function5.h  Makefile    sample2.cc  sample4.cc  sample5.o
function5.cc  function5.o  sample1.cc  sample3.cc  sample5.cc
[nukazuka@sphnx05 01:43:30 answers] $
[nukazuka@sphnx05 01:43:31 answers] $ g++ sample5.o function5.o
[nukazuka@sphnx05 01:43:48 answers] $ ls
a.out        function5.cc  function5.o  sample1.cc  sample3.cc  sample5.cc
function4.h  function5.h   Makefile     sample2.cc  sample4.cc  sample5.o
[nukazuka@sphnx05 01:43:49 answers] $
[nukazuka@sphnx05 01:43:51 answers] $ ./a.out
Input value: 2
Twice of 2 is 4
[nukazuka@sphnx05 01:43:53 answers] $
```

ファイルごとにコンパイルするので、必要なものだけコンパイルし直すことも可能。
大規模なプログラムでは必須。

C++ sample5: 資料

コンパイル方法 C: function5 から（共有）ライブラリを作成し、
sample5.cc コンパイル時にライブラリを読み込む。

```
[nukazuka@sphnx05 01:54:45 answers] $ ls
function4.h   function5.h      Makefile    sample2.cc  sample4.cc
function5.cc                   sample1.cc  sample3.cc  sample5.cc
[nukazuka@sphnx05 01:54:47 answers] $ g++ -shared -fPIC -o libfunction5.so function5.cc
[nukazuka@sphnx05 01:54:53 answers] $ ls
function4.h   function5.h      Makefile    sample2.cc  sample4.cc
function5.cc  libfunction5.so  sample1.cc  sample3.cc  sample5.cc
[nukazuka@sphnx05 01:54:55 answers] $
[nukazuka@sphnx05 01:54:56 answers] $ g++  -L. -lfunction5 sample5.cc
[nukazuka@sphnx05 01:55:03 answers] $ ls
a.out         function5.cc  libfunction5.so  sample1.cc  sample3.cc  sample5.cc
function4.h   function5.h   Makefile         sample2.cc  sample4.cc
[nukazuka@sphnx05 01:55:06 answers] $
[nukazuka@sphnx05 01:55:07 answers] $ ./a.out
Input value: 2
Twice of 2 is 4
```

共有ライブラリ作成に必要なオプション

共有ライブラリ名は
lib + 任意 + .so
とするのが普通

使用するライブラリを追加するオプション
-l + (共有ライブラリ名から lib と .so を取ったもの)とする

ライブラリ検索場所を追加するオプション
カレントディレクトリ . を追加している

広く使い回されるプログラムはライブラリとして作ることが多い。
Fun4All では自分の analysis module を共有ライブラリとして作成し、ROOT マクロで読み込んでいる。

C++ sample5 + α:
関数をヘッダーファイルとソースファイルに分けて書く

分割コンパイルは便利だが、手動は面倒
make コマンドで自動化してみる

make には Makefile で何をどう make するかルールを記述する必要がある

読み方:メイクコマンド

makeコマンド

概要 makeコマンドとは、UNIX系OSにおけるプログラム開発で標準的に用いられるコマンドの一つで、ソースコードからの実行ファイルの作成(ビルド)を自動化するもの。

目次
- 概要
- 関連用語
- 他の辞典の解説

ポスト

人間がプログラミング言語で書いたソースコードから実行可能なプログラムファイルを得るには、ソースコードをコンパイルして機械語などで書かれたオブジェクトコードのプログラムに変換し、複数のオブジェクトコードや外部のライブラリファイルなどを連結(リンク)して一つの実行ファイルにする作業が必要となる。

単純なプログラムではこの工程は数回のコマンド実行で済むが、プログラムの規模が大きくなり構成が複雑になると、多数のファイルをコンパイルしたりリンクしなければならず、手作業で行うのは面倒で誤りも起きがちになる。

makeコマンドは一連の手順を所定の形式でテキストファイル(Makefile)に記述しておくと、これに従ってコマンド実行などを連続して自動的に行なってくれる。makeコマンド一回の実行で実行ファイルの作成が完了する。

何度も繰り返しビルドを行う場合、各ファイルの最終更新日時を確認して前回のビルドから更新されたファイルだけを再コンパイルしたりリンクし直す機能を備えており、単に繰り返し同じコマンドを実行する場合よりも短時間で効率的に再ビルドすることができる。

初版は1976年にC言語によるプログラム開発を支援するために開発された。ある程度の汎用性があり、他の言語によるプログラムのビルドや、プログラム以外のファイル生成に応用することもできる。L多くのLinuxディストリビューションを含むUNIX系OSに標準で収録されている。

引用

C++ sample5: 資料

コンパイル方法 B: ソースファイルごとにコンパイルして、最後にまとめる（分割コンパイル）、愚直に書いてみる

```
10 ##############################
11 # The rule for sample5_2     #
12 #----------------------------#
13 # Do as follows:             #
14 #   $ make sample5_2         #
15 ##############################
16 sample5_2: sample5.o function5.o
17         g++ -o sample5_2 sample5.o function5.o
18
19 sample5.o:
20         g++ -c sample5.cc
21
22 function5.o:
23         g++ -c function5.cc
```

INTT_Fun4All_Tutorial/cpp_basics/answers/Makefile

1. sample5_2 は、sample5.o と function5.o に依存してると書いてある
2. sample5.o と function5.o を作る
3. それらを使って sample5_2 を作る

```
[nukazuka@sphnx05 03:03:21 answers] $ make sample5_2
g++ -c sample5.cc
g++ -c function5.cc
g++ -o sample5_2 sample5.o function5.o
[nukazuka@sphnx05 03:03:25 answers] $ ls
function4.h   function5.h  Makefile    sample2.cc  sample4.cc  sample5.cc  temp.cc
function5.cc  function5.o  sample1.cc  sample3.cc  sample5_2   sample5.o
[nukazuka@sphnx05 03:03:29 answers] $
[nukazuka@sphnx05 03:03:30 answers] $ ./sample5_2
Input value: 2
Twice of 2 is 4
```

C++ sample5: 資料

コンパイル方法 B: ソースファイルごとにコンパイルして、最後にまとめる（分割コンパイル）、スマートに書いてみる

```
25 ##############################
26 # The rule for sample5_2_2   #
27 #-----------------------------#
28 # Do as follows:              #
29 #   $ make sample5_2_2        #
30 ##############################
31 sample5_2_2: sample5.o function5.o
32         g++ -o $@ $^
33
34 # General rule to make *.o
35 %.o: %.cc
36         g++ -o $@ -c $<
37
```

ここはしょうがない

$@ : 作るものの名前に置換される
$^ : 依存するファイル名に置換される
$< : 最初の依存するファイル名に置換される
%: ワイルドカード（任意の文字列）

INTT_Fun4All_Tutorial/cpp_basics/answers/Makefile

```
[nukazuka@sphnx05 03:12:37 answers] $ make sample5_2_2
g++ -c sample5.cc
g++ -c function5.cc
g++ -o sample5_2_2 sample5.o function5.o
[nukazuka@sphnx05 03:12:46 answers] $ ls
function4.h   function5.h  Makefile    sample2.cc  sample4.cc   sample5.cc  temp.cc
function5.cc  function5.o  sample1.cc  sample3.cc  sample5_2_2  sample5.o
[nukazuka@sphnx05 03:12:48 answers] $ ./sample5_2_2
Input value: 2
Twice of 2 is 4
```

C++ sample5: 資料

コンパイル方法 C: function5 から（共有）ライブラリを作成し、sample5.cc コンパイル時にライブラリを読み込む。

```
##############################
# The rule for sample5_3     #
#----------------------------#
# Do as follows:             #
#   $ make sample5_3         #
##############################
#sample5_3: sample5.o libfunction5.so
#       g++ -o $@ $^
sample5_3: libfunction5.so
        g++ -o $@ -L. -lfunction5 sample5.cc

libfunction5.so: function5.cc
        g++ -shared -fPIC -o $@ $^

```

INTT_Fun4All_Tutorial/cpp_basics/answers/Makefile

```
[nukazuka@sphnx05 03:16:18 answers] $ make sample5_3
g++ -shared -fPIC -o libfunction5.so function5.cc
g++ -o sample5_3 -L. -lfunction5 sample5.cc
[nukazuka@sphnx05 03:17:13 answers] $
[nukazuka@sphnx05 03:17:16 answers] $ ls
function4.h   function5.h      Makefile    sample2.cc  sample4.cc  sample5.cc
function5.cc  libfunction5.so  sample1.cc  sample3.cc  sample5_3   temp.cc
[nukazuka@sphnx05 03:17:16 answers] $ ./sample5_3
Input value: 2
Twice of 2 is 4
```

sample 2

Making an analysis module

I. You are at Fun4All_samples. Move to sample_module_2

```
$ cd sample_module_2
```

II. Make a build directory

```
$ mkdir build
```

build directory contains files generated during compilation to avoid confusion.

III. Make an install directory

```
$ mkdir install
```

The library files and the header files are put in a certain directory. You can set the path to this install directory to ROOT_INCLUDE_PATH and LD_LIBRARY_PATH variables to use them.

IV. Move to the build directory

```
$ cd build
```

sample 2

Making an analysis module

V. Generating the configuration files by autogen.sh

```
$ autogen.sh --prefix=${PWD}/../install
```

prefix option specifies where the libraries and the headers are installed. You need to give an absolute path, but it's trouble. You can use an environmental variable ${PWD}, which is an absolute path of the current directory.

```
[nukazuka@sphnx05 13:53:28 build] $ ../autogen.sh  --prefix=${PWD}/../install
libtoolize: putting auxiliary files in `.'.
libtoolize: linking file `./ltmain.sh'
libtoolize: Consider adding `AC_CONFIG_MACRO_DIR([m4])' to configure.ac and
libtoolize: rerunning libtoolize, to keep the correct libtool macros in-tree.
libtoolize: Consider adding `-I m4' to ACLOCAL_AMFLAGS in Makefile.am.
configure: loading site script /cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/et
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether the C++ compiler works... yes
checking for C++ compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C++ compiler... yes
checking whether /cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c96
checking for style of include used by make... GNU
checking dependency style of /cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/
checking build system type... x86_64-unknown-linux-gnu
```

```
$ tre -l 2 -a
.
├── ltmain.sh
├── Makefile.am
├── Makefile.in
├── aclocal.m4
├── autogen.sh
├── config.guess
├── README.md~
├── config.sub
├── configure
├── configure.ac
├── depcomp
├── install-sh
├── missing
├── tutorial.cc
├── tutorial.h
├── README.md
├── build          ← You are here
│   ├── config.log
│   ├── testexternals.cc
│   ├── config.status
│   ├── testexternals.o
│   ├── Makefile
│   ├── .deps
│   ├── libtool
│   ├── tutorial.lo
│   ├── libtutorial.la
│   └── testexternals
├── install
└── autom4te.cache
```

sample 2

Making your own analysis module

```
[nukazuka@sphnx05 13:53:58 build] $ ls
config.log  config.status  libtool  Makefile
```

VI. compiling the analysis module by make

```
$ make
```

```
[nukazuka@sphnx05 13:55:11 build] $ make
echo "//*** this is a generated file. Do not commit, do not edit" > testexternals.cc
echo "int main()" >> testexternals.cc
echo "{" >> testexternals.cc
echo "  return 0;" >> testexternals.cc
echo "}" >> testexternals.cc
make  all-am
make[1]: Entering directory `/gpfs/mnt/gpfs02/sphenix/user/nukazuka/repo/INTT_Fun4All_Tutorial/ver2024/Fun4All_sample
/bin/sh ./libtool  --tag=CXX   --mode=compile /cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c9
AGE_TARNAME=\"tutorial\" -DPACKAGE_VERSION=\"1.00\" -DPACKAGE_STRING=\"tutorial\ 1.00\" -DPACKAGE_BUGREPORT=\"\" -DPA
STDC_HEADERS=1 -DHAVE_SYS_TYPES_H=1 -DHAVE_SYS_STAT_H=1 -DHAVE_STDLIB_H=1 -DHAVE_STRING_H=1 -DHAVE_MEMORY_H=1 -DHAVE_
ISTD_H=1 -DHAVE_DLFCN_H=1 -DLT_OBJDIR=\".libs/\" -I. -I..  -I/sphenix/u/nukazuka/user/repo/INTT_Fun4All_Tutorial/ver2
e -I/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_new/new.10/include -isystem/cvmfs/sphenix.sdcc.bnl.gov/gcc
-std=c++17 -Wall -Werror -MT tutorial.lo -MD -MP -MF .deps/tutorial.Tpo -c -o tutorial.lo ../tutorial.cc
libtool: compile:  /cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c96/x86_64-centos7/bin/g++ -D
PACKAGE_VERSION=\"1.00\" "-DPACKAGE_STRING=\"tutorial 1.00\"" -DPACKAGE_BUGREPORT=\"\" -DPACKAGE_URL=\"\" -DPACKAGE=\
TYPES_H=1 -DHAVE_SYS_STAT_H=1 -DHAVE_STDLIB_H=1 -DHAVE_STRING_H=1 -DHAVE_MEMORY_H=1 -DHAVE_STRINGS_H=1 -DHAVE_INTTYPE
```

To check whether compiling was done successfully: `$ echo $?`

VII.installing the library (*.so) and the header file (*.h)

```
$ make install
```

sample 2

Making your own analysis module

VIII.installing the library (*.so) and the header file (*.h)

```
$ make install
```

```
[nukazuka@sphnx05 13:58:04 build] $ make install
make  install-am
make[1]: Entering directory `/gpfs/mnt/gpfs02/sphenix/user/nukazuka/repo/INTT_Fun4All_Tutorial/ver2024/Fun4All_samples/sample_mod
make[2]: Entering directory `/gpfs/mnt/gpfs02/sphenix/user/nukazuka/repo/INTT_Fun4All_Tutorial/ver2024/Fun4All_samples/sample_mod
 /usr/bin/mkdir -p '/sphenix/u/nukazuka/user/repo/INTT_Fun4All_Tutorial/ver2024/Fun4All_samples/sample_module_2/build/../install/
 /bin/sh ./libtool   --mode=install /usr/bin/install -c   libtutorial.la '/sphenix/u/nukazuka/user/repo/INTT_Fun4All_Tutorial/ver
nstall/lib'
libtool: install: /usr/bin/install -c .libs/libtutorial.so.0.0.0 /sphenix/u/nukazuka/user/repo/INTT_Fun4All_Tutorial/ver2024/Fun4
b/libtutorial.so.0.0.0
libtool: install: (cd /sphenix/u/nukazuka/user/repo/INTT_Fun4All_Tutorial/ver2024/Fun4All_samples/sample_module_2/build/../instal
orial.so.0 || { rm -f libtutorial.so.0 && ln -s libtutorial.so.0.0.0 libtutorial.so.0; }; })
libtool: install: (cd /sphenix/u/nukazuka/user/repo/INTT_Fun4All_Tutorial/ver2024/Fun4All_samples/sample_module_2/build/../instal
```

```
[nukazuka@sphnx05 13:59:51 build] $ tre -a ../install
../install
├── lib
│   ├── libtutorial.so.0.0.0
│   ├── libtutorial.so.0
│   ├── libtutorial.so
│   └── libtutorial.la
└── include
    └── tutorial
        └── tutorial.h
```

You need to inform the path to this directory to ROOT to use them.

sample 2

Analysis module

The standard way to implement the class, add it to the ROOT macro, and run it is

1. generating a template by CreateSubsysRecoModule.pl

```
$ CreateSubsysRecoModule.pl [name_of_the_module] [options]
```

Joseph's minimum example is also a good start.

2. generating the configuration files by autogen.sh

```
$ autogen.sh --prefix=[install_path]
```

3. implementing the header file (*.h) and the source file (*.cc) by yourself.
4. compiling the analysis module by make command

```
$ make
```

5. installing the library (*.so) and the header file (*.h)

```
$ make install
```

Let's use sample_module_2

Try them

HANDS ON! #5

sample 2

Making your own analysis module

6. setting your `LD_LIBRARY_PATH` and `ROOT_INCLUDE_PATH`

   `LD_LIBRARY_PATH`: an environmental variable generally used in Linux to find libraries

   `ROOT_INCLUDE_PATH`: an environmental variable introduced by ROOT to find header files

   The basic setup of them is done by `/opt/sphenix/core/bin/sphenix_setup.sh`.

```
[nukazuka@sphnx05 14:02:43 build] $ echo $ROOT_INCLUDE_PATH | tr : "\n"
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/macros/detectors/sPHENIX
./
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/incl
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/incl
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/event_
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/event_
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmics/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmics/install/include/in
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmics/install/include/an
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmics/install/include/in
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/calibrations/intt/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/calibrations/intt/install/include/intt
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/calibrations/intt/install/include/intt
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/offline/packages/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/offline/packages/install/include/intt
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/calibrations/intt/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/calibrations/intt/install/include/int
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/inclu
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/inclu
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/inclu
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/correl
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/correl
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/timing
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/timing
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/timing
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/QAhtml_repo/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/QAhtml_repo/install/include/qahtml
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/QA_codes/install/include
```

```
[nukazuka@sphnx05 14:03:34 build] $ echo $LD_LIBRARY_PATH | tr : "\n"
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/genki/Fun4All_codes/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/event_
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/general_codes/Jaein/Cosmics/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/calibrations/intt/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/offline/packages/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware2/calibrations/intt/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware_latest/offline/framework/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/correl
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/analysis/INTT_preliminary/202409_performance/timing
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/QAhtml_repo/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/INTT/QA_codes/install/lib
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/InttEventDisplay/install/lib
/sphenix/user/nukazuka/gamma_jet/analysis/MC/MC_truth/install/lib
/sphenix/user/nukazuka/gamma_jet/analysis/MC/mbd/install/lib
```

Files in the paths in the variables can be used with only the file name.

Here, sed_path command is defined by Genki:

```
function sed_path ()
{
    echo $@ | sed -e "s/:/\n/g"
}
```

sample 2

Making your own analysis module

6. setting your `LD_LIBRARY_PATH` and `ROOT_INCLUDE_PATH`

   To add paths to the variables, you can run /opt/sphenix/core/bin/setup_local.sh, for example,

```
$ source /opt/sphenix/core/bin/setup_local.sh [absolute path to your install directory] [another if you want]
...
```

```
[nukazuka@sphnx04 03:49:44 ~] $ sed_path $LD_LIBRARY_PATH
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/simulation/g4simulation/g4intt/install/lib
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c96/x86_64-centos7/lib
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/gcc/12.1.0-57c96/x86_64-centos7/lib64
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/opt/sphenix/core/binutils/2.37-355ed/x86_64-centos7/lib
.
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/r
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/r
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/o
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/o
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/o
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/o
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/o
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/o
/afs/rhic.bnl.gov/app/insure-7.5.5/lib
/usr/local/lib64
/usr/lib64
```

```
[nukazuka@sphnx04 03:49:51 ~] $ sed_path $ROOT_INCLUDE_PATH
./
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/simulation/g4simulation/g4intt/install/include
/sphenix/tg/tg01/commissioning/INTT/work/genki/repos/coresoftware/simulation/g4simulation/g4intt/install/include/g4intt
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/ffarawobjects
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/JSON
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/half
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/torch
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g4detectors
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/eventplane
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/kineto
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g4decayer
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/phfield
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/LHAPDF
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/c10
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/oneapi
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/DDCond
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/g4tracking
/cvmfs/sphenix.sdcc.bnl.gov/gcc-12.1.0/release/release_ana/ana.382/include/litecaloeval
```

sample 2

Making your own analysis module

6. setting your `LD_LIBRARY_PATH` and `ROOT_INCLUDE_PATH`

   Typing the command every time is trouble. You should write it to ${HOME}/.bashrc, so the command is executed just after login. For example,

```
test.sh x +
#!/bin_bash

source /opt/sphenix/core/bin/sphenix_setup.sh
source /opt/sphenix/core/bin/setup_local.sh /sphenix/tg/tg01/commissioning/INTT/wo\
rk/genki/repos/coresoftware/simulation/g4simulation/g4intt/install
```

   You can add multiple paths as arguments with separation with a space.

   To do it in a more user-friendly way, you can use my script:
   `/sphenix/tg/tg01/commissioning/INTT/repositories/libraries/intt_setup_v2.sh`
   An explanation of this script can be found in the backup slide (planned).

sample 2

Making your own analysis module

7. adding an include statement and `R__LOAD_LIBRARY` macro to your ROOT macro. and execute it!

```
[nukazuka@sphnx05 14:21:49 Fun4All_samples] $ root -q -b Fun4All_minimum_2.C

Processing Fun4All_minimum_2.C...
tutorial::tutorial(const std::string &name) Calling ctor
tutorial::Init(PHCompositeNode *topNode) Initializing
Fun4AllServer::setRun(): run 0 uses CDB TIMESTAMP 0
tutorial::InitRun(PHCompositeNode *topNode) Initializing for Run XXX
-------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
   RUN (PHCompositeNode)/
   PAR (PHCompositeNode)/


tutorial::process_event(PHCompositeNode *topNode) Processing Event
tutorial::ResetEvent(PHCompositeNode *topNode) Resetting internal structures, prepare for next event
tutorial::EndRun(const int runnumber) Ending Run for Run 0
tutorial::End(PHCompositeNode *topNode) This is the End...
tutorial::Reset(PHCompositeNode *topNode) being Reset
tutorial::~tutorial() Calling dtor
```

Write 2 lines to ${HOME}/.bashrc →

```
test.sh x | +
1 #!/bin_bash
2
3 source /opt/sphenix/core/bin/sphenix_setup.sh
4 source /opt/sphenix/core/bin/setup_local.sh /sphenix/tg/tg01/commissioning/INTT/wo\
  rk/genki/repos/coresoftware/simulation/g4simulation/g4intt/install
```

Change the path to your install directory's path.

6. setting your `LD_LIBRARY_PATH` and `ROOT_INCLUDE_PATH`
7. run Fun4All_minimum_2.C

Try them

```
[nukazuka@sphnx05 14:21:49 Fun4All_samples] $ root -q -b Fun4All_minimum_2.C

Processing Fun4All_minimum_2.C...
tutorial::tutorial(const std::string &name) Calling ctor
tutorial::Init(PHCompositeNode *topNode) Initializing
Fun4AllServer::setRun(): run 0 uses CDB TIMESTAMP 0
tutorial::InitRun(PHCompositeNode *topNode) Initializing for Run XXX
-------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
   RUN (PHCompositeNode)/
   PAR (PHCompositeNode)/


tutorial::process_event(PHCompositeNode *topNode) Processing Event
tutorial::ResetEvent(PHCompositeNode *topNode) Resetting internal structures, prepare for next event
tutorial::EndRun(const int runnumber) Ending Run for Run 0
tutorial::End(PHCompositeNode *topNode) This is the End...
tutorial::Reset(PHCompositeNode *topNode) being Reset
tutorial::~tutorial() Calling dtor
```

HANDS ON! #6

sample 2

Practical example

It depends on what you want to do. For example:

- inputting raw file(s)
- ✓ inputting DST file(s)
- Monte-Carlo as an input

- ✓ running someone's analysis codes ← but it does nothing…
- running your analysis codes

- Outputting results to DST file(s)
- Outputting results to histograms/TTrees

This super simple macro takes no input file and outputs nothing. 1 event is processed.

DST

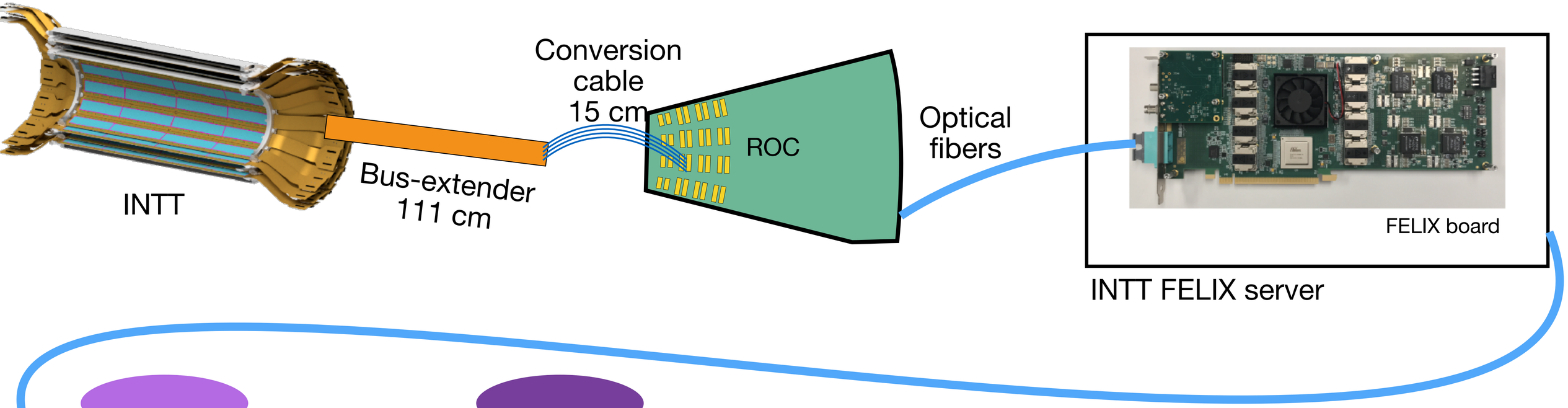
What do we touch in our analysis module?

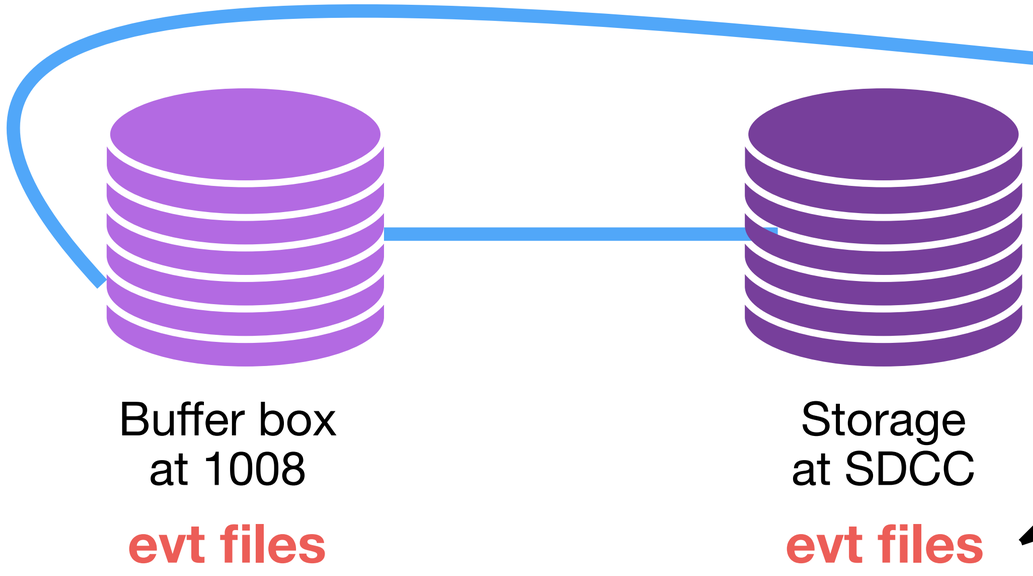
In a DST file, you can touch data through so-called “node”. Data is given as an object of a class.

```
Data: /sphenix/lustre01/sphnxpro/physics/slurp/streaming/physics/inttonlyrun_00051100_00051200/DST_INTT_EVENT_run2pp_new_2024p002-00051171-00000.root

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      GL1 (PHCompositeNode)/
         GL1RAWHIT (IO,Gl1Packetv2)
      INTT (PHCompositeNode)/
         INTTRAWHIT (IO,InttRawHitContainerv2) ← Data of raw hits of INTT
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
   PAR (PHCompositeNode)/
```

DST

Data process flow and data type





Private/Special DST files

`/sphenix/tg/tg01/commissioning/INTT/data`

We are no longer recommended to use private DSTs.

Special DSTs due to technical difficulties are OK.

Offical DST files

`/sphenix/lustre01/sphnxpro/physics/slurp/`

Please use them if you don't have technical issues.

There are a lot of types of DSTs. Chaotic…

DST

Official DSTs

See here for more details

```
/sphenix/lustre01/sphnxpro/
├── bbox3
├── db_bkup
├── testbed
├── beam
├── ml
├── bbox1
├── mlp
├── cosmics
├── 1008_backup
├── physics
├── bbox5
├── bbox2
├── slurp
├── calib
├── slurptest
├── mdc2
├── tchou
├── HPSS_Status
├── run1auau_pi0_calib
├── t1044
├── bbox4
├── production
├── liumigrate
├── bbox0
├── commissioning
├── scratch
└── sdcc
```

Data taken in the early phase were processed, and their DSTs are put here.

```
/sphenix/lustre01/sphnxpro/physics
├── tpc
├── INTT
├── GL1
├── run2pp
├── mbd
├── ZDC
├── MVTX
├── slurp
├── online_monitoring
├── run2auau
├── mbdcalib
├── TPOT
├── LL1
├── emcal
├── HCal
└── TEST
```

SLURP (sPHENIX Ligtweight Utilities for Realtime Production)

```
/sphenix/lustre01/sphnxpro/physics/slurp/
├── calocosmics
├── jetproduction
├── cosmics
├── caloy2test
├── tpccosmics
├── caloy2calib
├── streaming
│   ├── cosmics
│   ├── physics
│   └── fast
├── fast_tracking
├── calobeam
├── calophysics
├── tracking
│   ├── run_00050900_00051000
│   ├── run_…
├── caloy2fitting
├── tpccalib
├── tpcbeam
├── junkdrawer
└── TEST
```

DST

Contents of the official DSTs: InttRawHit

XXXXXXXX: run number
YYYYY: segment number

- Path: /sphenix/lustre01/sphnxpro/physics/slurp/streaming/physics/inttonlyrun_*
  - file name: DST_INTT_EVENT_run2pp_new_2024p002-XXXXXXXX-YYYYY.root
  - Contents:

```
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      GL1 (PHCompositeNode)/
         GL1RAWHIT (IO,Gl1Packetv2)
      INTT (PHCompositeNode)/
         INTTRAWHIT (IO,InttRawHitContainerv2)
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
   PAR (PHCompositeNode)/
```

InttRawHit is a class for raw hits of INTT.
Parameters are almost same as those in testbench.

Public Member Functions

	InttRawHit ()=default	
virtual	~InttRawHit ()=default	
virtual uint64_t	get_bco () const	•GTM BCO 40 bits unsigned integer
virtual void	set_bco (const uint64_t)	
virtual int32_t	get_packetid () const	•Packet ID, i.e. FELIX server ID 3001—3008
virtual void	set_packetid (const int32_t)	
virtual uint32_t	get_word () const	•word
virtual void	set_word (uint32_t)	
virtual uint16_t	get_fee () const	•ID of fee (front end electronics), which means a half-ladder (0—13)
virtual void	set_fee (uint16_t)	
virtual uint16_t	get_channel_id () const	•ID of the channel (silicon strip) 0—127
virtual void	set_channel_id (uint16_t)	
virtual uint16_t	get_chip_id () const	•ID of the chip (silicon chip) 1—26
virtual void	set_chip_id (uint16_t)	
virtual uint16_t	get_adc () const	•ADC value 0—7
virtual void	set_adc (uint16_t)	
virtual uint16_t	get_FPHX_BCO () const	•FPHX BCO value 0—127
virtual void	set_FPHX_BCO (uint16_t)	
virtual uint16_t	get_full_FPHX () const	•for debugging
virtual void	set_full_FPHX (uint16_t)	
virtual uint16_t	get_full_ROC () const	•for debugging
virtual void	set_full_ROC (uint16_t)	
virtual uint16_t	get_amplitude () const	•amplitude of calibration pulse 0—63
virtual void	set_amplitude (uint16_t)	

sPHENIX Doxygen > InttRawHit

Contents of the official DSTs: InttRawHit with other tracking detectors

- Path:/sphenix/lustre01/sphnxpro/physics/slurp/streaming/physics/run_*
  - file name: DST_STREAMING_EVENT_run2pp_new_2024p002-XXXXXXXX-YYYYY.root
  - Contents:

```
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      GL1 (PHCompositeNode)/
         GL1RAWHIT (IO,Gl1Packetv2)
      INTT (PHCompositeNode)/
         INTTRAWHIT (IO,InttRawHitContainerv2)
      MVTX (PHCompositeNode)/
         MVTXRAWEVTHEADER (IO,MvtxRawEvtHeaderv2)
         MVTXRAWHIT (IO,MvtxRawHitContainerv1)
      MICROMEGAS (PHCompositeNode)/
         MICROMEGASRAWHIT (IO,MicromegasRawHitContainerv1)
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
   PAR (PHCompositeNode)/
```

DST

DST

Contents of the official DSTs: TrkrHit

- Path:/sphenix/lustre01/sphnxpro/physics/slurp/tracking/new_2024p007/run_*
  - file name: DST_TRKR_HIT_run2pp_new_2024p007-XXXXXXXX-YYYYY.root
  - Contents:

```
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
      TRKR (PHCompositeNode)/
         TRKR_HITSET (IO,TrkrHitSetContainerv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
      CYLINDERGEOM_MVTX (IO,PHObject)
      CYLINDERGEOM_INTT (IO,PHObject)
      CYLINDERCELLGEOM_SVTX (IO,PHObject)
      CYLINDERGEOM_MICROMEGAS_FULL (IO,PHObject)
      GEOMETRY_IO (IO,PHObject)
      CdbUrl (IO,CdbUrlSavev1)
   PAR (PHCompositeNode)/
```

Public Member Functions

	~TrkrHit () override dtor
void	identify (std::ostream &os=std::cout) const override
void	Reset () override Clear Event.
int	isValid () const override isValid returns non zero if object contains vailid data
virtual void	addEnergy (const double)
virtual double	getEnergy ()
virtual void	setAdc (const unsigned int)
virtual unsigned int	getAdc ()

TrkrHit

If you want to analyze hits but not clusters without hot channels and clone hits, it can be a choice.

InttRawHit
Hits from other tracking detectors

Our starting point

hot channel rejection
BCO filter
clone hits removal (not on purpose)

TrkrHit → TrkrCluster

clustering

For most of the studies at a higher level, you can use it.

DST

Contents of the official DSTs: TrkrCluster

- Path:/sphenix/lustre01/sphnxpro/physics/slurp/tracking/new_2024p007/run_*
  - file name: DST_TRKR_CLUSTER_run2pp_new_2024p007-XXXXXXXX-YYYYY.root
  - Contents:

```
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
      TRKR (PHCompositeNode)/
         TRKR_CLUSTER (IO,TrkrClusterContainerv4)
         TRKR_CLUSTERCROSSINGASSOC (IO,TrkrClusterCrossingAssocv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
      CYLINDERGEOM_MVTX (IO,PHObject)
      CYLINDERGEOM_INTT (IO,PHObject)
      CYLINDERCELLGEOM_SVTX (IO,PHObject)
      CYLINDERGEOM_MICROMEGAS_FULL (IO,PHObject)
      GEOMETRY_IO (IO,PHObject)
      CdbUrl (IO,CdbUrlSavev1)
   PAR (PHCompositeNode)/
```

TrkrCluster

	~TrkrCluster () override=default dtor
void	Identify (std::ostream &os=std::cout) const override
void	Reset () override Clear Event.
int	IsValid () const override isValid returns non zero if object contains vailid data
virtual void	CopyFrom (const TrkrCluster &) copy content from base class
virtual void	CopyFrom (TrkrCluster *) copy content from base class
virtual float	getLocalX () const
virtual void	setLocalX (float)
virtual float	getLocalY () const
virtual void	setLocalY (float)
virtual void	setAdc (unsigned int)
virtual unsigned int	getAdc () const
virtual void	setMaxAdc (uint16_t)
virtual unsigned int	getMaxAdc () const
virtual char	getOverlap () const
virtual void	setOverlap (char)
virtual char	getEdge () const
virtual void	setEdge (char)
virtual void	setTime (const float)
virtual float	getTime () const
virtual char	getSize () const
virtual float	getPhiSize () const
virtual float	getZSize () const
virtual float	getPhiError () const
virtual float	getRPhiError () const
virtual float	getZError () const
virtual void	setActsLocalError (unsigned int, unsigned int, float) Acts functions, for Acts modules use only.
virtual float	getActsLocalError (unsigned int, unsigned int) const

InttRawHit
Hits from other tracking detectors

→ hot channel rejection
BCO filter
clone hits removal (not on purpose)

→ TrkrHit

→ clustering

→ TrkrCluster

Contents of the official DSTs: SiliconTrackSeedContainer

- Path:/sphenix/lustre01/sphnxpro/physics/slurp/tracking/run_*
  - File name: DST_TRKR_SEED_run2pp_new_2024p007-XXXXXXXX-YYYYY.root
  - Contents

```
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
      SVTX (PHCompositeNode)/
         SiliconTrackSeedContainer (IO,PHObject)
         TpcTrackSeedContainer (IO,PHObject)
         SvtxTrackSeedContainer (IO,PHObject)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
      CYLINDERGEOM_MVTX (IO,PHObject)
      CYLINDERGEOM_INTT (IO,PHObject)
      CYLINDERCELLGEOM_SVTX (IO,PHObject)
      CYLINDERGEOM_MICROMEGAS_FULL (IO,PHObject)
      GEOMETRY_IO (IO,PHObject)
      CdbUrl (IO,CdbUrlSavev1)
   PAR (PHCompositeNode)/
```

DST

SiliconTrackSeed implemented by the tracking group is the tracking results using MVTX + INTT.
SvtxTrackSeed uses all tracking detectors to reconstruct tracks.
But I haven't touched them yet…

sample 3

Sample3: Analyzing InttRawHit

Fun4All_minimum_3.C

```c
#include <fun4all/Fun4AllServer.h>

//#include <fun4all/SubsysReco.h>

#include <fun4all/Fun4AllInputManager.h>
#include <fun4all/Fun4AllDstInputManager.h>

R__LOAD_LIBRARY(libfun4all.so)
R__LOAD_LIBRARY(libfun4allraw.so)

// It should be tutorial.h of sample_moudle_3
#include <tutorial.h>
R__LOAD_LIBRARY( libtutorial.so )

int Fun4All_minimum_3( int nEvents = 1,
                       const string &data = "/sphenix/lustre01/sphnxpro/physics/sl\
urp/streaming/physics/inttonlyrun_00051100_00051200/DST_INTT_EVENT_run2pp_new_2024\
p002-00051171-00000.root" )
{

  Fun4AllServer *se = Fun4AllServer::instance();

  Fun4AllInputManager *in = new Fun4AllDstInputManager("DSTin");
  in->AddFile( data );
  se->registerInputManager(in);

  //in->fileopen( data );
  tutorial* analysis_module = new tutorial( "name" );
  analysis_module->Verbosity( 2 ); // 0: minimum(default), 1: event by event info,\
 2: hit by hit info
  se->registerSubsystem( analysis_module );

  se->run(nEvents);
  se->End();
  delete se;

  gSystem->Exit(0);
  return 0;
}
```

sample_module_3/tutorial.h
sample_module_3/tutorial.cc

Let's grab an official DST to analyze INTT raw hits.

- Use Fun4All_minimum_3.C and sample_module_3

← Path to the DST file to be used. Only file name is actually OK.

DST is read in a Fun4All macro. See next page.

Use analysis module #3
Tip: You can set the level of information to be printed on your terminal using `SubsysReco::Verbosity` function

sample 3

Reading a DST file

You need to read DST(s) in your Fun4All macro using Fun4AllInputManager.
You need to include some header files.

Including header files:

```
#include <fun4all/Fun4AllInputManager.h>
#include <fun4all/Fun4AllDstInputManager.h>
```

Single file:

```
Fun4AllInputManager *in = new Fun4AllDstInputManager("DSTin");
in->AddFile( data );
se->registerInputManager(in);
```

Single file (old method but event skipping works):

```
Fun4AllInputManager *in = new Fun4AllDstInputManager("DSTin");
in->fileopen( data );
se->registerInputManager(in);
```

Multiple files:

```
Fun4AllInputManager *in = new Fun4AllDstInputManager("DST");
in->AddListFile( list );
```

Saved

Genki Nukazuka 3 months ago
Hi, which function should I use to read DST files?

- `Fun4AllInputManager::AddFile`
- `Fun4AllInputManager::AddListFile`
- `Fun4AllDstInputManager::fileopen`

7 replies

Anthony Hodges 3 months ago
In a given Fun4AllMacro, you'll probably instantiate a Fun4AllInputManager, which uses AddFile for a single DST, and AddListFile for a file list containing a single column of DST files, such as that generated by CreateDstList.pl

Christopher Pinkenburg 3 months ago
::fileopen is an older API, but you need it when you want to skip events

Genki Nukazuka 3 months ago
Thank you, @Anthony Hodges and @Christopher Pinkenburg !
So, if I want to skip some events, I shouldn't use AddFile and AddListFile. Is that right?

Christopher Pinkenburg 3 months ago
yes - that only works with fileopen() - the reason is that fileopen actually opens the DST and only then Fun4All can set up the skipping. Fixing this is a long standing item on my list but it's not that straightforward so that'll be a while

Genki Nukazuka 3 months ago
Thank you for the explanation. OK, that's good to know. So, the recommendation is

- for a single file: Fun4AllDstInputManager::fileopen
- for multiple files: Fun4AllInputManager::AddListFile

Is it right?

Christopher Pinkenburg 3 months ago
yes (where if you don't need the skipping, I would go for the AddFile() - it shares the underlying machinery with AddListFile() so if you go from single file to filelist it'll go through the same channels)

Genki Nukazuka 3 months ago
Great! Thanks a lot!

Discussion in Mattermost

back to ToC

sample 3

Analysis module #3: sample_module_3/tutorial.h

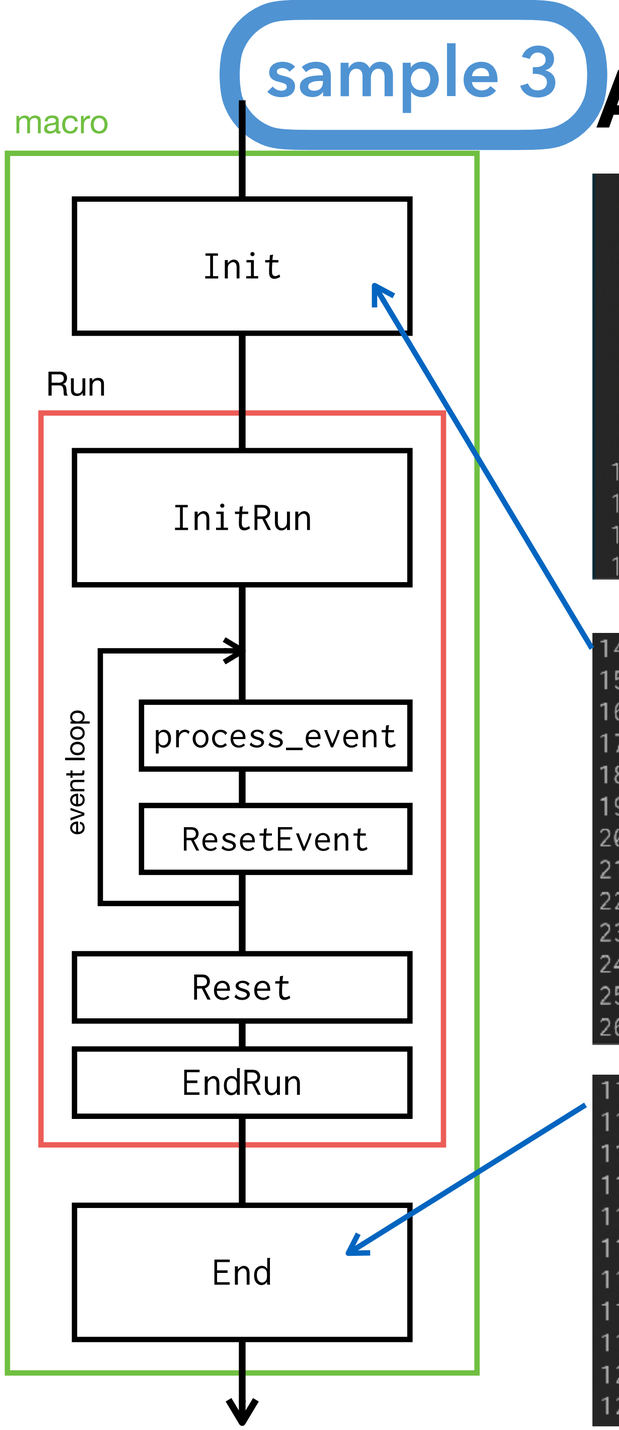
```cpp
// Tell emacs that this is a C++ source
//  -*- C++ -*-.
#ifndef TUTORIAL_H
#define TUTORIAL_H

// Fun4All libraries
#include <fun4all/SubsysReco.h>
#include <fun4all/Fun4AllReturnCodes.h>
#include <phool/PHCompositeNode.h>
#include <phool/getClass.h>

#include <ffarawobjects/InttRawHit.h>
#include <ffarawobjects/InttRawHitv2.h>
#include <ffarawobjects/InttRawHitContainer.h>
#include <ffarawobjects/InttRawHitContainerv2.h>

#include <string>
#include <iostream>
#include <iomanip>

// ROOT libraries
#include <TFile.h>
#include <TH1D.h>
#include <TFile.h>

class PHCompositeNode;
```

```cpp
class tutorial : public SubsysReco
{
 public:

  tutorial(const std::string &name = "tutorial");

  ~tutorial() override;

  //! Init function of this analysis module. ROOT file is opened. A histgram object is created.
  int Init(PHCompositeNode *topNode) override;

  int InitRun(PHCompositeNode *topNode) override;

  //! This function is executed in each event
  int process_event(PHCompositeNode *topNode) override;

  /// Clean up internals after each event.
  int ResetEvent(PHCompositeNode *topNode) override;

  /// Called at the end of each run.
  int EndRun(const int runnumber) override;

  /// Called at the end of all processing.
  int End(PHCompositeNode *topNode) override;

  /// Reset
  int Reset(PHCompositeNode * /*topNode*/) override;

  void Print(const std::string &what = "ALL") const override;

  //! You can set the name of the output file, otherwise it's tutorial_sample3.root
  void SetOutputPath( std::string path ){ output_path_ = path; };

 private:

  // private member variables and objects for output
  std::string output_path_ = "tutorial_sample3.root";
  TFile* output_; //! I/O of output ROOT file
  TH1D* hist_hit_num_; //! a histogram for the number of raw hit per event
  void PrintHitParameterHeader(); //! Printing a header line for raw hit parameters
};

#endif // TUTORIAL_H
```

← The actual analysis codes are written in this function.

for output

Analysis module #3: sample_module_3/tutorial.cc




```
1  #include "tutorial.h"
2
3  tutorial::tutorial(const std::string &name):
4    SubsysReco(name),
5    output_( nullptr ),
6    hist_hit_num_( nullptr )
7  {
8  }
9
10 tutorial::~tutorial()
11 {
12 }
13
```

Contractor and destructor

```
14 int tutorial::Init(PHCompositeNode *topNode)
15 {
16
17   ////////////////////////////////////////////////////////
18   // Initialization of the member                       //
19   ////////////////////////////////////////////////////////
20   output_ = new TFile( output_path_.c_str(), "RECREATE" );
21
22   hist_hit_num_ = new TH1D( "hit_num", "Number of raw hit distribution;#Cluster;Entries",
23                             10000, 0, 10000 );
24
25   return Fun4AllReturnCodes::EVENT_OK;
26 }
```

Init function

← ROOT file preparation for output

← Histogram preparation

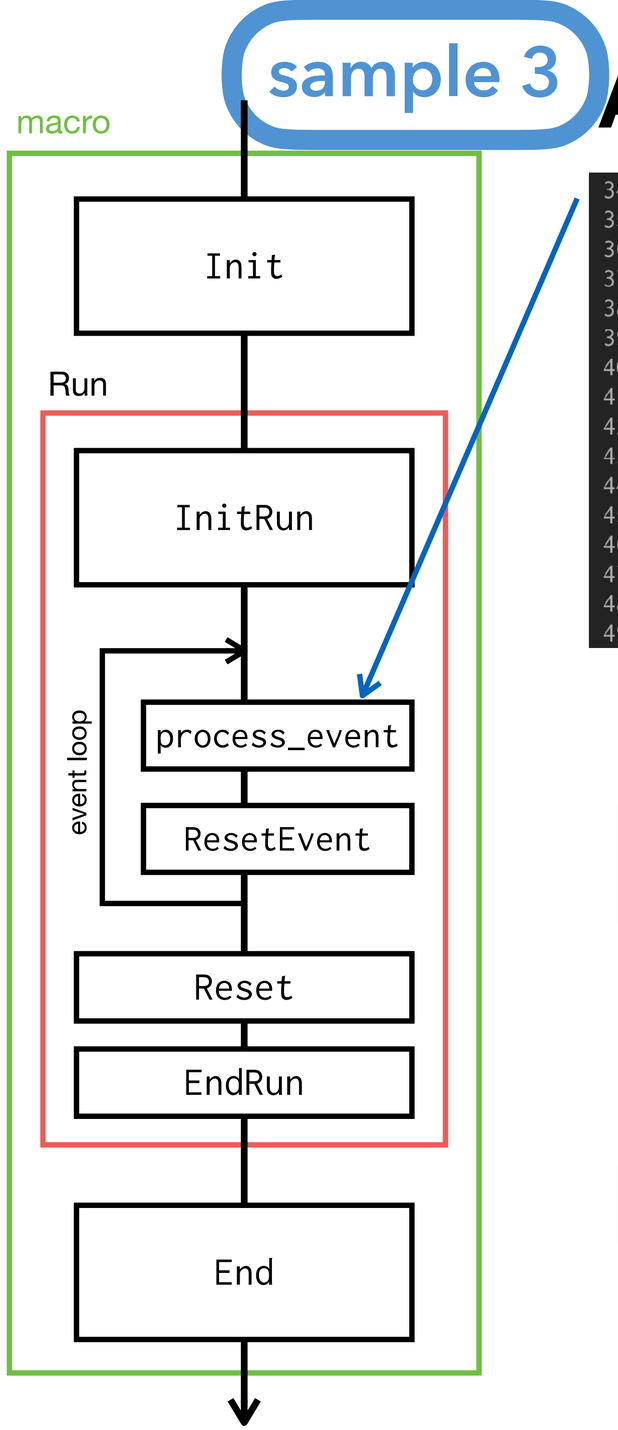
```
111 int tutorial::End(PHCompositeNode *topNode)
112 {
113
114   ////////////////////////////////////////////////////////
115   // Writing objects to the output file                 //
116   ////////////////////////////////////////////////////////
117   output_->WriteTObject( hist_hit_num_, hist_hit_num_->GetName() );
118   output_->Close();
119
120   return Fun4AllReturnCodes::EVENT_OK;
121 }
```

End function

← Writing the histogram to the output ROOT file

← Closing the output ROOT file

Analysis module #3: sample_module_3/tutorial.cc




```
int tutorial::process_event(PHCompositeNode *topNode)
{

  ////////////////////////////////////////////////////////////////////////////////////////
  // Getting INTTRAWHIT node                                                            //
  ////////////////////////////////////////////////////////////////////////////////////////
  std::string node_name_inttrawhit = "INTTRAWHIT";
  InttRawHitContainer* node_inttrawhit_map_ =
    findNode::getClass<InttRawHitContainer>(topNode, node_name_inttrawhit);

  if (!node_inttrawhit_map_)
    {
      std::cerr << PHWHERE << node_name_inttrawhit << " node is missing." << std::endl;
      return Fun4AllReturnCodes::ABORTEVENT;
    }

```

← If InttRawHitContainer node is not found, it's better to skip this event for safety.

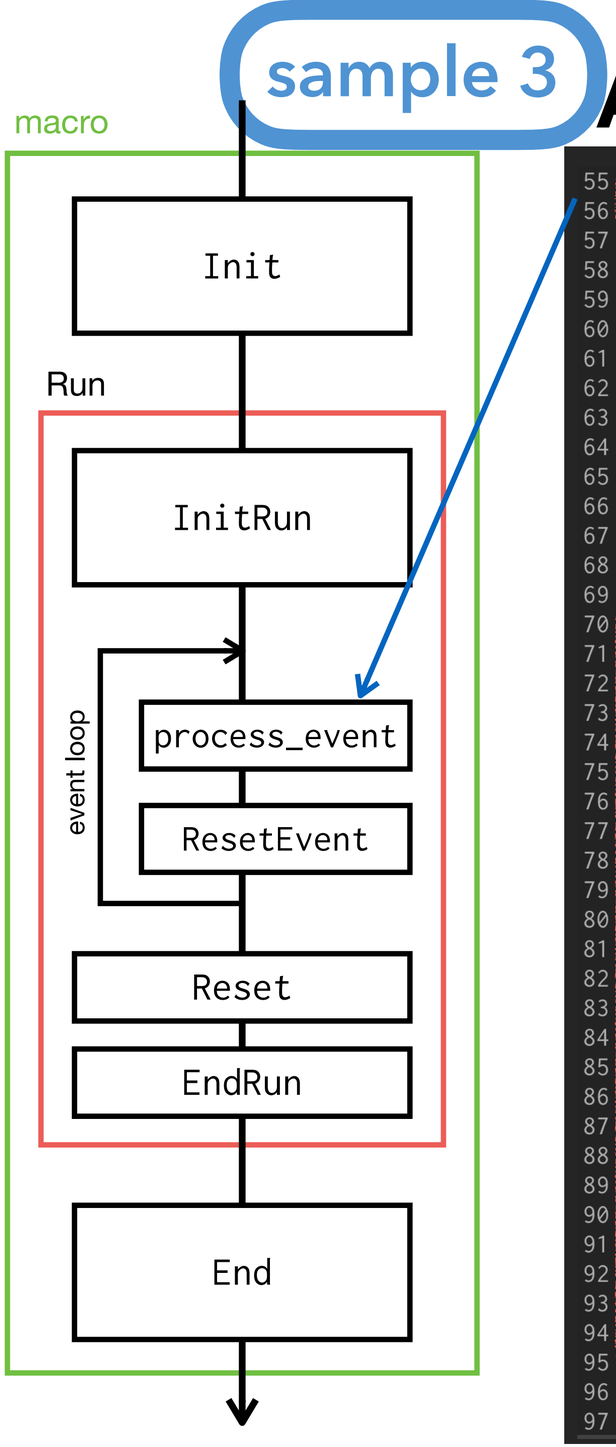
To access InttRawHit, you need to get an InttRawHitContainer.

```
InttRawHitContainer* node_inttrawhit_map_ =
  findNode::getClass<InttRawHitContainer>(topNode, node_name_inttrawhit);
```

Through the InttRawHitContainer, you can get InttRawHit (see next page)

```
72        InttRawHit* hit = node_inttrawhit_map_->get_hit(i);
```

Analysis module #3: sample_module_3/tutorial.cc




```
55  int raw_hit_num = node_inttrawhit_map_->get_nhits();
56  hist_hit_num_->Fill( raw_hit_num );
57
58  /// If an user wants to see detailed informaiton, print it out
59
60  if( this->Verbosity() > 0 )
61    {
62      if( this->Verbosity() > 1 )
63        {
64          std::cout << std::string( 100, '-' ) << std::endl;
65        }
66
67      std::cout << "#raw hit: " << std::setw(5) << raw_hit_num << std::endl;
68    }
69
70  for (unsigned int i = 0; i < raw_hit_num; i++)
71    {
72      InttRawHit* hit = node_inttrawhit_map_->get_hit(i);
73
74      if( this->Verbosity() > 1 )
75        {
76
77          if( i % 30 == 0 )
78            this->PrintHitParameterHeader();
79
80          std::cout
81            << std::setw( 10 ) << hit->get_event_counter() << " " // uint32_t
82            // << std::setw() << hit->get_word() << " " // uint32_t
83            << std::setw(  5 ) << hit->get_packetid() << " " // int32_t
84            << std::setw( 14 ) << hit->get_fee() << " " // uint16_t
85            << std::setw(  4 ) << hit->get_chip_id() << " " // uint16_t
86            << std::setw(  4 ) << hit->get_channel_id() << " " // uint16_t
87            << std::setw(  3 ) << hit->get_adc() << " " // uint16_t
88            << std::setw( 20 ) << hit->get_bco() << " " // uint64_t
89            << std::setw(  8 ) << hit->get_FPHX_BCO() << " " // uint16_t
90            // hit->get_full_FPHX(); // uint16_t, for low level debugging
91            // hit->get_full_ROC(); // uint16_t, for low level debugging
92            // hit->get_amplitude(); // uint16_t, for calibration data
93            << std::endl;
94        }
95    }
96  return Fun4AllReturnCodes::EVENT_OK;
97 }
```

getting the number of raw hits in this event and filling the histogram with it

for loop over all InttRawHit

← If you set the verbosity level higher than 1, parameters are printed on you terminal

these are all methods to access raw hit's parameter

sample 3

Analysis module #3: sample_module_3/tutorial.cc

Example1 (Verbosity Lv. 0)

```
[nukazuka@sphnx06 02:00:38 work_now] $ time root -q -b 'Fun4All_minimum_3.C( 10 ) '

Processing Fun4All_minimum_3.C( 10 ) ...
Fun4AllServer::setRun(): run 51171 uses CDB TIMESTAMP 51171
------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      GL1 (PHCompositeNode)/
         GL1RAWHIT (IO,Gl1Packetv2)
      INTT (PHCompositeNode)/
         INTTRAWHIT (IO,InttRawHitContainerv2)
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
   PAR (PHCompositeNode)/
```

Example2 (Verbosity Lv. 1)

```
[nukazuka@sphnx06 02:00:26 work_now] $ time root -q -b 'Fun4All_minimum_3.C( 10 ) '

Processing Fun4All_minimum_3.C( 10 ) ...
Fun4AllServer::setRun(): run 51171 uses CDB TIMESTAMP 51171
------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      GL1 (PHCompositeNode)/
         GL1RAWHIT (IO,Gl1Packetv2)
      INTT (PHCompositeNode)/
         INTTRAWHIT (IO,InttRawHitContainerv2)
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
   PAR (PHCompositeNode)/


#raw hit:  1670
#raw hit:  1734
#raw hit:  1510
#raw hit:  1368
#raw hit:  1110
#raw hit:  1336
#raw hit:  3068
#raw hit:  1009
#raw hit:  1202
#raw hit:  1303
```

Example3 (Verbosity Lv. 2)

```
[nukazuka@sphnx06 01:11:57 work_now] $ time root -q -b 'Fun4All_minimum_3.C( 10 ) '

Processing Fun4All_minimum_3.C( 10 ) ...
Fun4AllServer::setRun(): run 51171 uses CDB TIMESTAMP 51171
------------------------------------

List of Nodes in Fun4AllServer:
Node Tree under TopNode TOP
TOP (PHCompositeNode)/
   DST (PHCompositeNode)/
      GL1 (PHCompositeNode)/
         GL1RAWHIT (IO,Gl1Packetv2)
      INTT (PHCompositeNode)/
         INTTRAWHIT (IO,InttRawHitContainerv2)
      Sync (IO,SyncObjectv1)
      EventHeader (IO,EventHeaderv1)
   RUN (PHCompositeNode)/
      RunHeader (IO,RunHeaderv1)
      Flags (IO,FlagSavev1)
   PAR (PHCompositeNode)/


-----------------------------------------------------------------------------------------------------
#raw hit:  1670
Event cntr PktID Half Ladder ID Chip Chan ADC              GTM BCO FPHX BCO
         1  3001              0    8  111   6          527098524396        5
         1  3001              0    1   71   6          527098524396       27
         1  3001              0    1   72   7          527098524396       27
         1  3001              0    1   80   5          527098524396       27
         1  3001              0    9   70   7          527098524396       54
         1  3001              0   17    3   5          527098524396      109
```

sample 3

Analysis module #3: Hands on

HANDS ON! #7

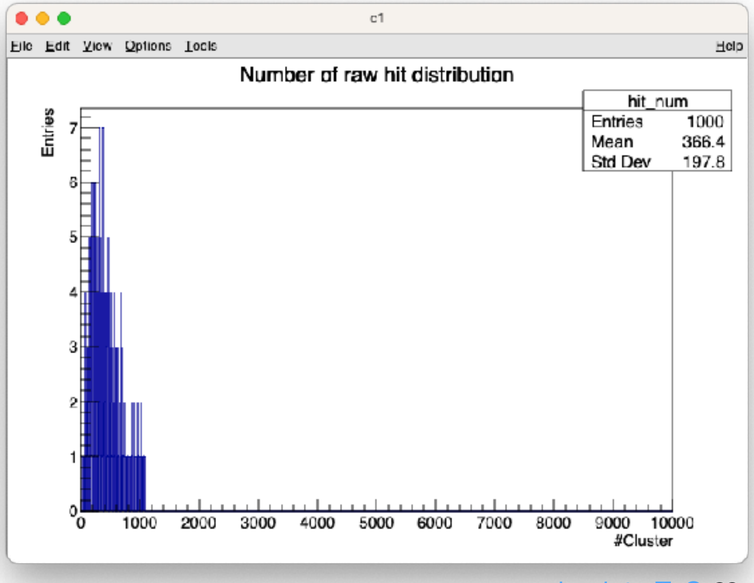
Let's try

1. Compile and install sample_module_3

```
1.1. $ mv sample_module_3
1.2. $ mkdir build
1.3. $ mkdir install
1.4. $ cd build
1.5. $ ../autogen.sh --prefix=$PWD/../install
1.6. $ make install
```

2. Modify your environmental variable
   - 2.1. ROOT_INCLUDE_PATH
   - 2.2. LD_LIBRARY_PATH
3. Run Fun4All_minimum_3.C with verbosity level 0, 1, and 2

```
3.1. Verbosity level 0: analysis_module->Verbosity( 0 );
     $ root -q -b  'Fun4All_minimum_3.C(1000)'
3.2. Verbosity level 1: analysis_module->Verbosit ( 1 );
     $ root -q -b  'Fun4All_minimum_3.C(1000)'
3.3. Verbosity level 2: analysis_module->Verbosity( 2 );
     $ root -q -b  'Fun4All_minimum_3.C(1000)'
```



sample 4

Sample4: Analyzing TrkrCluster

Fun4All_minimum_4.C

```
// Fun4All headers
#include <fun4all/Fun4AllServer.h>
#include <fun4all/Fun4AllDstInputManager.h>

// Some general header macros
#include <GlobalVariables.C>
#include <G4Setup_sPHENIX.C>
#include <G4_Input.C>

// Trkr headers
#include <Trkr_RecoInit.C>
#include <Trkr_Clustering.C>
#include <G4_ActsGeom.C>

// something else
#include <ffamodules/FlagHandler.h>
#include <ffamodules/HeadReco.h>
#include <ffamodules/SyncReco.h>
#include <ffamodules/CDBInterface.h>

#include <phool/PHRandomSeed.h>
#include <phool/recoConsts.h>

R__LOAD_LIBRARY(libfun4all.so)

#include <tutorial.h>
R__LOAD_LIBRARY( libtutorial.so )
```

Only the first part for including headers are shown.

Let's grab an official DST to analyze TrkrCluster.

- Use Fun4All_minimum_4.C and sample_module_4

The main difference between this Fun4All macro to the previous is the use of Acts geometry. It's needed to get cluster position in the lab frame.

sample 4

Sample4: Analyzing TrkrCluster

Fun4All_minimum_4.C

```c
// Fun4All headers
#include <fun4all/Fun4AllServer.h>
#include <fun4all/Fun4AllDstInputManager.h>

// Some general header macros
#include <GlobalVariables.C>
#include <G4Setup_sPHENIX.C>
#include <G4_Input.C>

// Trkr headers
#include <Trkr_RecoInit.C>
#include <Trkr_Clustering.C>
#include <G4_ActsGeom.C>

// something else
#include <ffamodules/FlagHandler.h>
#include <ffamodules/HeadReco.h>
#include <ffamodules/SyncReco.h>
#include <ffamodules/CDBInterface.h>

#include <phool/PHRandomSeed.h>
#include <phool/recoConsts.h>

R__LOAD_LIBRARY(libfun4all.so)

#include <tutorial.h>
R__LOAD_LIBRARY( libtutorial.so )
```

Only the first part for including headers are shown.

Another point here is that you need to include some .C files which are not in coresoftware repository. They are in macros repository.

macros / detectors / sPHENIX /

pinkenburg do not use quotes for include

Name

..

DisplayOn.C

Fun4All_G4_sPHENIX.C

G4Setup_sPHENIX.C

init_gui_vis.mac

vis.mac

Link

macros / common /

osbornjd Merge pull request #99

Name

..

CDBUtils.C

Calo_Calib.C

Calo_Fitting.C

DisplayOn.C

Fun4All_CalcProduction.C

G4_ActsGeom.C

G4_BeamLine.C

G4_BlackHole.C

G4_CEmc_Albedo.C

G4_CEmc_Spacal.C

G4_CaloTrigger.C

G4_Centrality.C

G4_DSTReader.C

Link

sample 4

Sample4: Analyzing TrkrCluster

Fun4All_minimum_4.C

```
// Fun4All headers
#include <fun4all/Fun4AllServer.h>
#include <fun4all/Fun4AllDstInputManager.h>

// Some general header macros
#include <GlobalVariables.C>
#include <G4Setup_sPHENIX.C>
#include <G4_Input.C>

// Trkr headers
#include <Trkr_RecoInit.C>
#include <Trkr_Clustering.C>
#include <G4_ActsGeom.C>

// something else
#include <ffamodules/FlagHandler.h>
#include <ffamodules/HeadReco.h>
#include <ffamodules/SyncReco.h>
#include <ffamodules/CDBInterface.h>

#include <phool/PHRandomSeed.h>
#include <phool/recoConsts.h>

R__LOAD_LIBRARY(libfun4all.so)

#include <tutorial.h>
R__LOAD_LIBRARY( libtutorial.so )
```

Only the first part for including headers are shown.

Another point here is that you need to include some .C files which are not in coresoftware repository. They are in macros repository.
There are 2 solutions:

1. Copy all files from the repository to the directory where this Fun4All macro is. I guess everyone in sPHENIX takes this way.
2. Setting the path to the directories in the repository to ROOT_INCLUDE_PATH. setup_local.sh cannot do it. You need to do it by yourself.

sample 4

Sample4: Analyzing TrkrCluster

Fun4All_minimum_4.C

```
int Fun4All_minimum_4(
                      int nEvents = 10,
                      const string &data = "/sphenix/lustre01/sphnxpro/physics/slurp/tr
                      )
{

  Fun4AllServer *se = Fun4AllServer::instance();
  //se->Verbosity(0);

  // Read DST
  Fun4AllInputManager *in = new Fun4AllDstInputManager("DSTin");
  in->fileopen( data );
  // in->AddListFile(inputfile); // to read a list of files, use it. A path to the text
  se->registerInputManager( in );

  // Flag Handler is always needed to read flags from input (if used)
  // and update our rc flags with them. At the end it saves all flags
  // again on the DST in the Flags node under the RUN node
  FlagHandler *flag = new FlagHandler();
  se->registerSubsystem(flag);
  Enable::CDB = true;
  // global tag

  recoConsts *rc = recoConsts::instance();
  rc->set_StringFlag("CDB_GLOBALTAG",CDB::global_tag);
  // 64 bit timestamp
  rc->set_uint64Flag("TIMESTAMP",CDB::timestamp);
  rc->set_IntFlag("RUNNUMBER", 0 );

  ////////////////////////////////////////////////////////////////////////////////
  // Something depends on Acts should be below....
  ////////////////////////////////////////////////////////////////////////////////
  // central tracking
  Enable::MVTX                                = true;
  Enable::TPC                                 = true;
  Enable::MICROMEGAS                          = true;
  Enable::INTT                                = true;
  Enable::BLACKHOLE                           = true;
  G4MAGNET::magfield_rescale = 1.4;

  // Initialize the selected subsystems
  // G4Init();

  // GEANT4 Detector description
  // if (!Input::READHITS)
  //   {
  //     G4Setup();
  //   }

  TrackingInit(); // necessary for ActsGeometry
```

This part is needed to get the cluster position in the lab frame. These are preparation for Acts geometry, which is mainly used for track reconstruction.

sample 4

Sample4: Analyzing TrkrCluster

Fun4All_minimum_4.C

```
80   ///////////////////////////
81   // Your analysis module //
82   ///////////////////////////
83   tutorial* analysis_module = new tutorial( "name" );
84   analysis_module->Verbosity( 1 ); // 0: minimum(default), 1: e
85   se->registerSubsystem( analysis_module );
86
87   //se->skip(skip);
88   se->run(nEvents);
89   se->End();
90
91   delete se;
92
93   gSystem->Exit(0);
94   return 0;
95 }
```

Your analysis module is generated and assigned here

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.h

```
// Tell emacs that this is a C++ source
//  -*- C++ -*-.
#ifndef TUTORIAL_H
#define TUTORIAL_H

// Fun4All libraries
#include <fun4all/SubsysReco.h>
#include <fun4all/Fun4AllReturnCodes.h>

//#include <ffaobjects/FlagSavev1.h>
#include <ffaobjects/EventHeaderv1.h>

#include <phool/PHCompositeNode.h>
#include <phool/getClass.h>
#include <phool/recoConsts.h>

#include <trackbase/ActsGeometry.h>
#include <trackbase/TrkrDefs.h>
#include <trackbase/TrkrClusterv4.h>
#include <trackbase/TrkrClusterContainerv4.h>

// std libraries
#include <string>
#include <iostream>
#include <iomanip>
#include <vector>

// ROOT libraries
#include "TFile.h"
#include "TTree.h"

class PHCompositeNode;
```

← for Acts geometry (libtrack.so is needed)

← for TrkrCluster

← for TrkrCluster

The first part of tutorial.h. Lots of header files are included than before.

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.h

```
class tutorial : public SubsysReco
{
 public:

  tutorial(const std::string &name = "tutorial");

  ~tutorial() override;

  int Init(PHCompositeNode *topNode) override;

  int InitRun(PHCompositeNode *topNode) override;

  int process_event(PHCompositeNode *topNode) override;

  /// Clean up internals after each event.
  int ResetEvent(PHCompositeNode *topNode) override;

  /// Called at the end of each run.
  int EndRun(const int runnumber) override;

  /// Called at the end of all processing.
  int End(PHCompositeNode *topNode) override;

  /// Reset
  int Reset(PHCompositeNode * /*topNode*/) override;

  void Print(const std::string &what = "ALL") const override;

  //! You can set the name of the output file, otherwise it's tutorial_sample4.root
  void SetOutputPath( std::string path ){ output_path_ = path; };
```

Public members in tutorial class.
Functions are same as sample #3.

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.h

Private members in tutorial class.
Many variables and 2 TTrees are added for output.

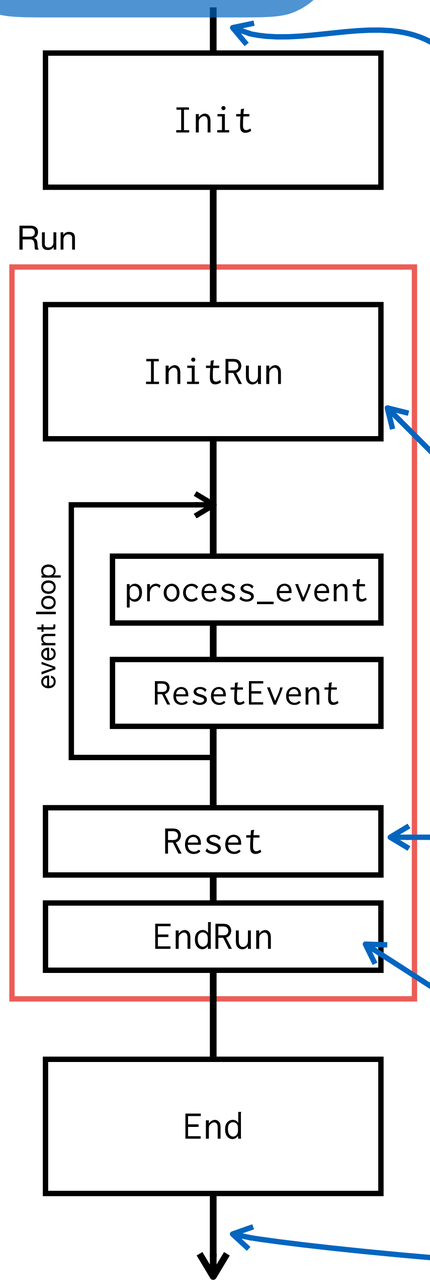
```
private:
 //! A function for the analysis of INTT clusters
 int cluster_analysis(PHCompositeNode *topNode, TrkrClusterContainerv4* node_cluster_map, ActsGeometry* node_acts );

 //! Reset function for cluster parameters
 int ResetClusterLoop(); ← Initialize/Reset variables of cluster-base TTree.

 std::string output_path_ = "tutorial_sample4.root";
 TFile* output_;        //! I/O of output ROOT file
 TTree* tree_event_;    //! Tree for event informaiton    ← TTree for event information. It's written to the output ROOT file.
 TTree* tree_cluster_; //! Tree for cluster information ← TTree for cluster information. It's written to the output ROOT file.

 //variables for tree_event_
 int run_num_ = 0;                    //! run number
 int event_id_ = 0;                   //! event number in this run
 int cluster_num_ = 0;                //! the number of clusters on INTT
 int cluster_num_layer_[4] = { 0 };   //! the number of clusters on each INTT layer (0-3)

 // variables for tree_cluster_
 float position_[3];                  //! cluster position in the lab-frame in cm
 int layer_ = 0;                      //! INTT layer ID for this cluster
 float adc_ = 0;                      //! ADC of this cluster ( not 0, 1, .., 7 but DAC value)
 float size_phi_ = 0;                 //! cluster size in phi direction
 float phi_ = 0;                      //! phi position of this cluster (radian)
 float theta_ = 0;                    //! theta position of this cluster (radian)
 float eta_ = 0;                      //! pseudorapidity of this cluster;

};

#endif // TUTORIAL_H
```

variables for event-base TTree

variables for cluster-base TTree

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc



Constructor and destructor

```
#include "tutorial.h"

tutorial::tutorial(const std::string &name):
 SubsysReco(name)
{}

tutorial::~tutorial()
{}
```

Functions which do nothing

```
int tutorial::InitRun(PHCompositeNode *topNode)
{

  return Fun4AllReturnCodes::EVENT_OK;
}
```

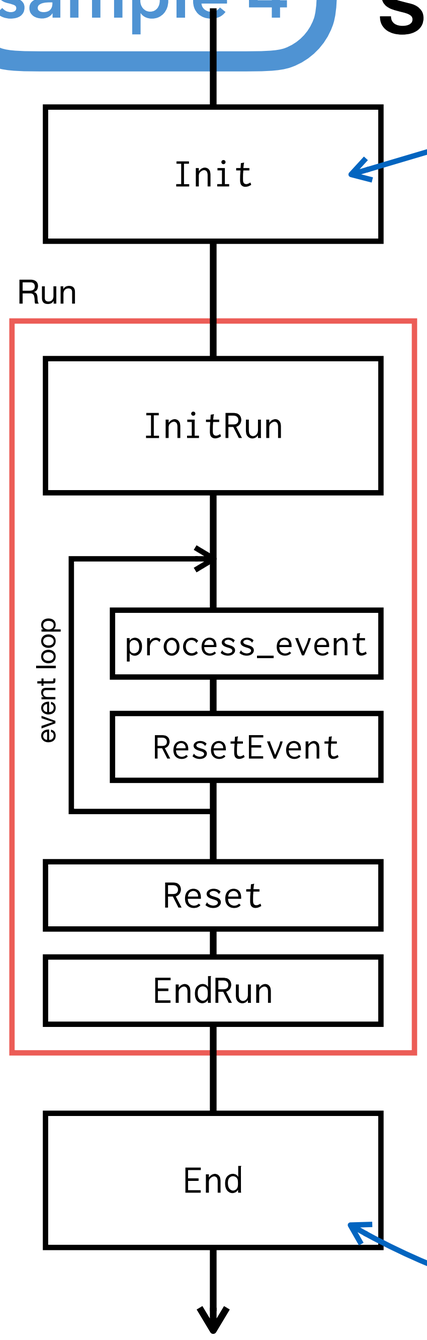
```
int tutorial::Reset(PHCompositeNode *topNode)
{

  return Fun4AllReturnCodes::EVENT_OK;
}
```

```
int tutorial::EndRun(const int runnumber)
{

  return Fun4AllReturnCodes::EVENT_OK;
}
```

```
void tutorial::Print(const std::string &what) const
{

}
```

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc




```cpp
int tutorial::Init(PHCompositeNode *topNode)
{

  ////////////////////////////////////////////////////////
  // Initialization of the member                       //
  ////////////////////////////////////////////////////////
  output_ = new TFile( output_path_.c_str(), "RECREATE" );

  // Preparation of event base tree
  tree_event_ = new TTree( "tree_event", "Event base TTree" );
  tree_event_->Branch( "run",         &run_num_,     "run_num/I" );
  tree_event_->Branch( "event",       &event_id_,    "event_id/I" );
  tree_event_->Branch( "cluster_num", &cluster_num_, "cluster_num/I" );

  // 0: inner layer of inner barrel
  // 1: outer layer of inner barrel
  // 2: inner layer of outer barrel
  // 3: outer layer of outer barrel
  tree_event_->Branch( "cluster_num_layer", &cluster_num_layer_, "cluster_num_layer_[4]/I" );

  // Preparation of cluster base tree
  tree_cluster_ = new TTree( "tree_cluster", "Cluster base TTree" );

  tree_cluster_->Branch( "run",      &run_num_,  "run_num/I" );
  tree_cluster_->Branch( "event",    &event_id_, "event_id/I" );
  tree_cluster_->Branch( "position", &position_, "position[3]/F" );
  tree_cluster_->Branch( "layer",    &layer_,    "layer/I" );
  tree_cluster_->Branch( "adc",      &adc_,      "adc/F" );
  tree_cluster_->Branch( "size_phi", &size_phi_, "size_phi_/F" );
  tree_cluster_->Branch( "phi",      &phi_,      "phi/F" );
  tree_cluster_->Branch( "theta",    &theta_,    "theta/F" );
  tree_cluster_->Branch( "eta",      &eta_,      "eta/F" );

  // Execute the reset function to assign initial value
  this->ResetEvent( topNode );
  return Fun4AllReturnCodes::EVENT_OK;
}
```

← Preparation of the output ROOT file

Preparation of
for event-base TTree

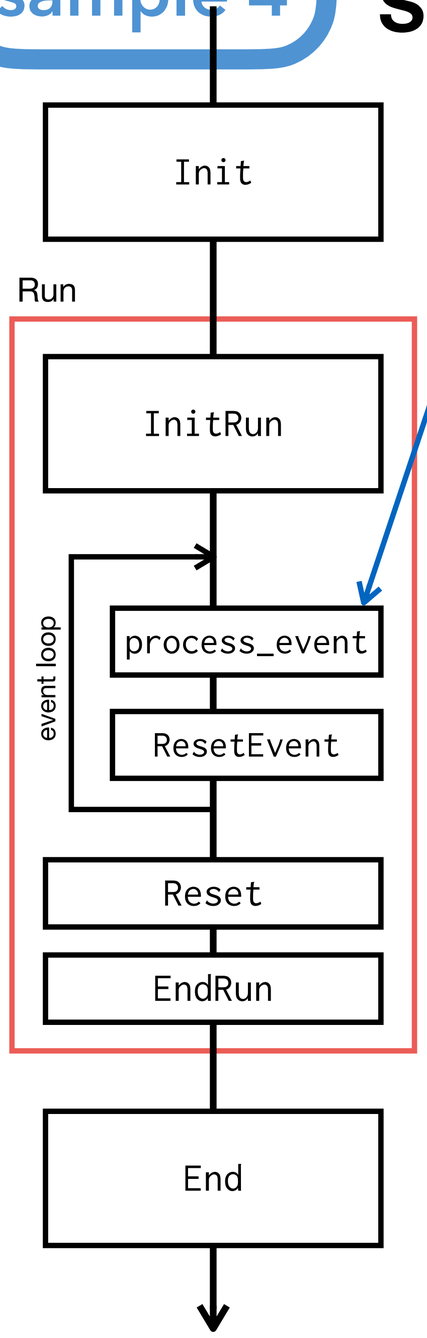
Preparation of
for cluster-base TTree

Trees are written into the ROOT file,
and the ROOT file is closed.

```cpp
int tutorial::End(PHCompositeNode *topNode)
{
  // Save Trees into the output file
  output_->WriteTObject( tree_event_, tree_event_->GetName() );
  output_->WriteTObject( tree_cluster_, tree_cluster_->GetName() );
  output_->Close();

  return Fun4AllReturnCodes::EVENT_OK;
}
```

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc




```
int tutorial::process_event(PHCompositeNode *topNode)
{

  ////////////////////////////////////////////////////////////////////////////////////////
  // Get nodes                                                                          //
  ////////////////////////////////////////////////////////////////////////////////////////

  //-----------------------------------------------//
  // Getting TrkrClusterContainer node             //
  // TRKR_CLUSTER node: Information of TrkrCluster  //
  //-----------------------------------------------//
  std::string node_name_trkr_cluster = "TRKR_CLUSTER";
  TrkrClusterContainerv4* node_cluster_map =
    findNode::getClass<TrkrClusterContainerv4>(topNode, node_name_trkr_cluster);

  if(!node_cluster_map )
    {
      std::cerr << PHWHERE << node_name_trkr_cluster << " node is missing." << std::endl;
      return Fun4AllReturnCodes::ABORTEVENT;
    }

  //-------------------------------------------------------------//
  // Getting Acts node to assign (x, y, z) coordinate to clusters //
  // ActsGeometry node: for the global coordinate                 //
  //-------------------------------------------------------------//
  ActsGeometry *node_acts = findNode::getClass<ActsGeometry>(topNode, "ActsGeometry");
  if ( !node_acts )
    {
      std::cout << PHWHERE << "No ActsGeometry on node tree. Bailing." << std::endl;
      return Fun4AllReturnCodes::ABORTEVENT;
    }

  //-------------------------------------//
  // Getting EventHeader to get event info //
  //-------------------------------------//
  EventHeaderv1* node_event_header = findNode::getClass<EventHeaderv1>( topNode, "EventHeader" );
  if( !node_event_header )
    {
      std::cout << PHWHERE << "No EventHeader on node tree. Skip this event." << std::endl;
      return Fun4AllReturnCodes::ABORTEVENT;
    }

```

variables
for event-base TTree

← Initialize/Reset variables of cluster-b

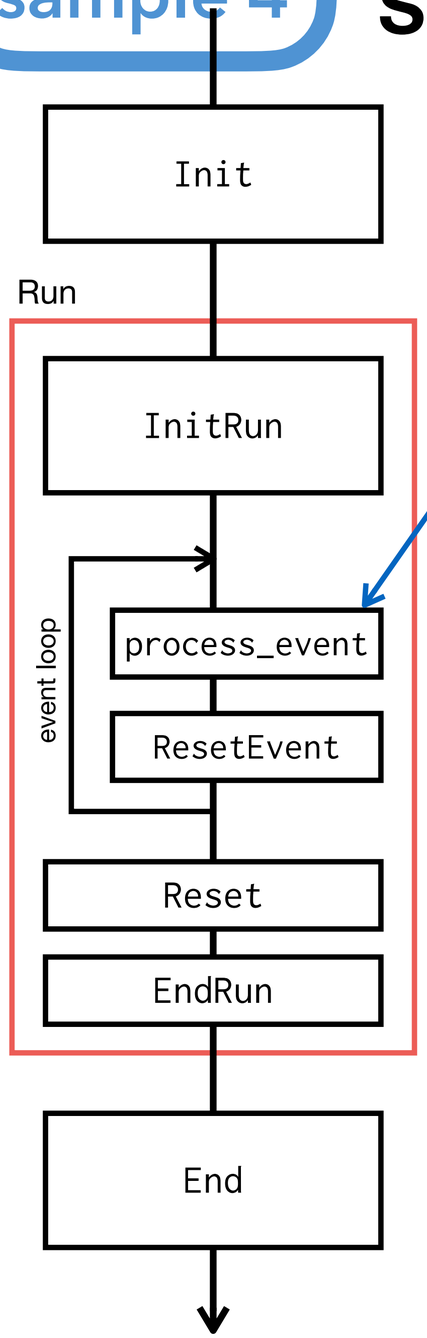
node for TrkrCluster

node for ActsGeometry

node for EventHeader

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc



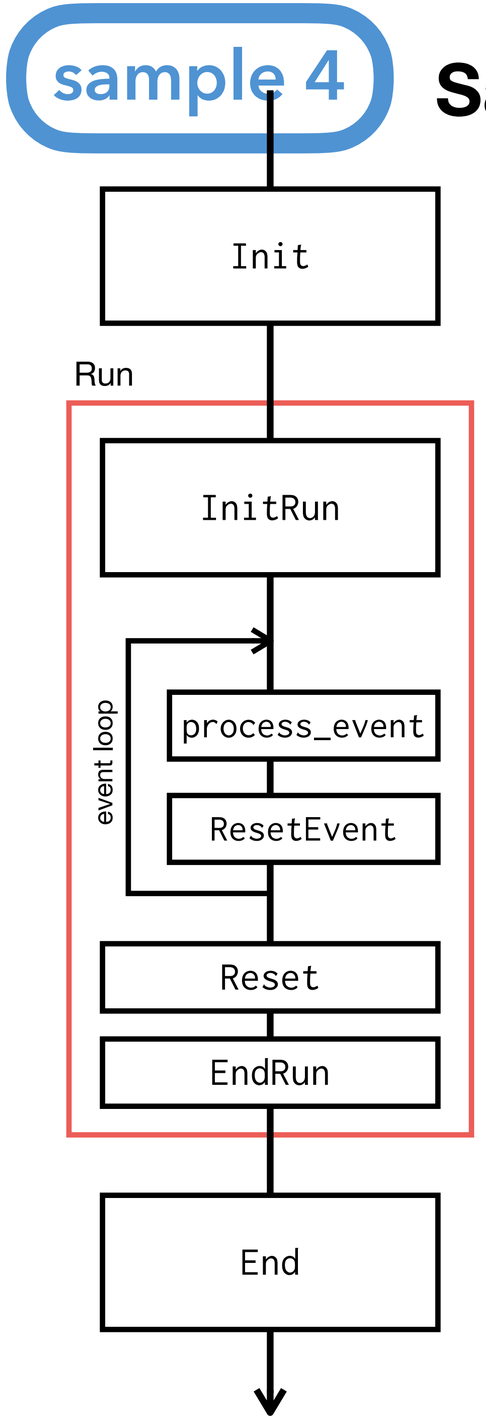

```
////////////////////////////////////////////////////////////////////////////////////////
// Analysis                                                                           //
////////////////////////////////////////////////////////////////////////////////////////
run_num_ = node_event_header->get_RunNumber();
event_id_ = node_event_header->get_EvtSequence();

// If user wants to see event infomation, do it
if( this->Verbosity() > 0 )
  {
    std::cout << "Run " << run_num_ << "\t"
              << "Event " << std::setw(10) << event_id_
              << std::endl;
  }

// analysis codes for INTT clusters are written in the function below
this->cluster_analysis( topNode, node_cluster_map, node_acts );

// Fill event-base TTree at the end of event process
tree_event_->Fill();
return Fun4AllReturnCodes::EVENT_OK;
}
```

Event information is taken from EventHeader and assigned to the variables

The part of cluster analysis is written in another function because it's long.

At the end of event process, the event-base TTree is filled.

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc



process_event

```
110   // analysis codes for INTT clusters are written in the function below
111   this->cluster_analysis( topNode, node_cluster_map, node_acts );
112
113   // Fill event-base TTree at the end of event process
114   tree_event_->Fill();
115   return Fun4AllReturnCodes::EVENT_OK;
116 }
```

```
118 int tutorial::cluster_analysis(PHCompositeNode *topNode, TrkrClusterContainerv4* node_cluster_map, ActsGeometry* node_acts )
119 {
120
121
122   // loop over all INTT layers (0: inner of inner barrel, 1: outer of inner, 2: inner of outer, 3: outer of outer)
123   for (unsigned int inttlayer = 0; inttlayer < 4; inttlayer++)
124     {
125
126       // get clusters only on the INTT layer, and loop over them
127       for (const auto &hitsetkey : node_cluster_map->getHitSetKeys(TrkrDefs::TrkrId::inttId, inttlayer + 3) )
128         {
129
130           // #cluster counters
131           cluster_num_++; // all of them
132           cluster_num_layer_[ inttlayer ]++; // for each layer
133
134           // type: std::pair<ConstIterator, ConstIterator> ConstRange
135           // here, MMap_::const_iterator ConstIterator;
136           auto range = node_cluster_map->getClusters(hitsetkey);
137
138           // loop over iterators of this cluster
139           for (auto clusIter = range.first; clusIter != range.second; ++clusIter)
140             {
141
142               const auto cluskey = clusIter->first;
143               const auto cluster = clusIter->second;
```

← For loop over 4 INTT layers

↑ For loop over clusters on this INTT layer

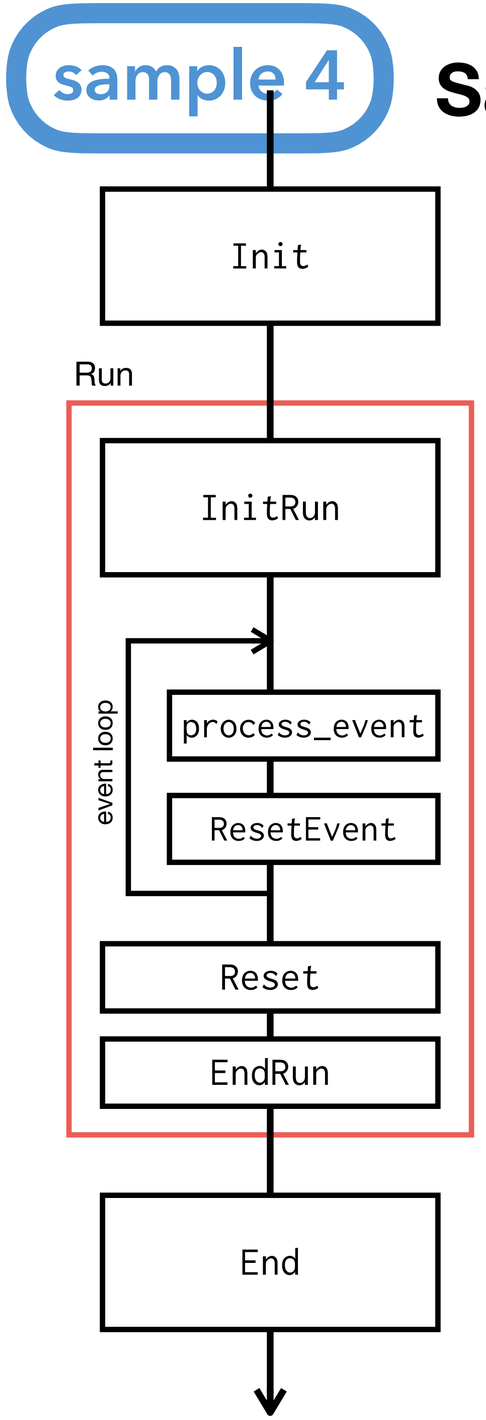
Counting #cluster

TrkrClusterContainer
→ pair or cluster key & TrkrCluster

cluster key

TrkrCluster

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc




```
145             // Get cluster position in lab-coordinate using Acts
146             const auto globalPos = node_acts->getGlobalPosition(cluskey, cluster);
147
148             // Set cluster position in lab-coordinate to t
149             cluster->setPosition(0,  globalPos.x() );
150             cluster->setPosition(1,  globalPos.y() );
151             cluster->setPosition(2,  globalPos.z() );
152
153             // Assign cluster parameters
154             position_[0] = cluster->getPosition( 0 ); // x
155             position_[1] = cluster->getPosition( 1 ); // y
156             position_[2] = cluster->getPosition( 2 ); // z
157             adc_ = cluster->getAdc();
158             size_phi_ = cluster->getPhiSize();
159
160             /** @TODO Calculate phi, theta, eta (pseudorapidity) by yourself
161               phi_ = 0; // (radian)
162               theta_ = 0; // (radian)
163               eta = 0; // pseudorapidity
164              */
165
166             // After getting all cluster parameters, let's fill them
167             tree_cluster_->Fill();
168
169             // Then, reset the parameters (it's not mandatory if all parameters are available all the time. It's just in case)
170             this->ResetClusterLoop();
```

↑ Getting cluster position in the lab frame by using Acts geometry. It's necessary because cluster key & TrkrCluster have only position in the INTT ladder.

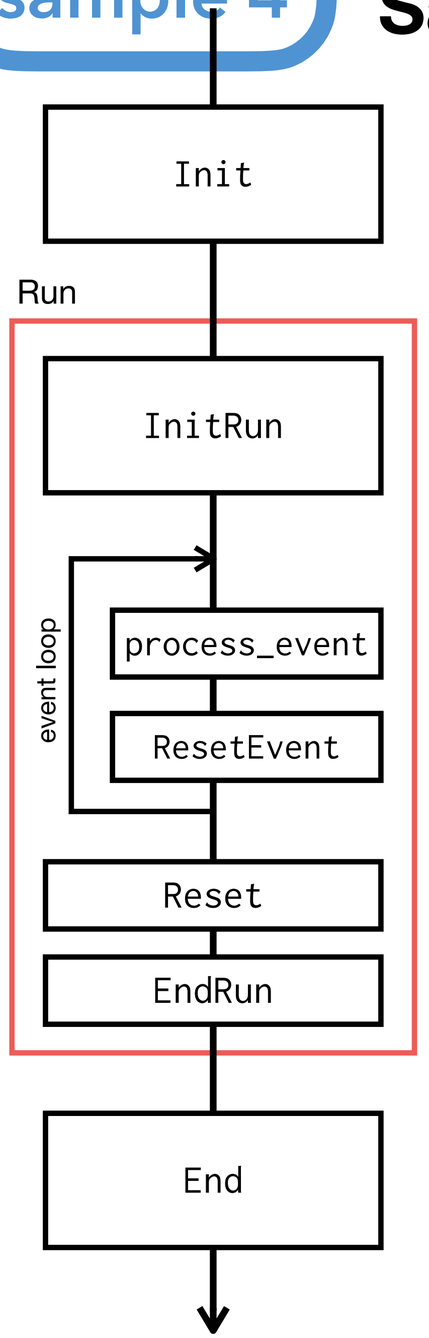
Getting cluster information and assigning them to the variables.

← Don't forget to fill cluster-base TTree

↑ The cluster variables are used to fill the cluster-base TTree and reused. If a value cannot be obtained for a cluster, the parameter value for the previous cluster is used. It's good to assign a default value before reusing.

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc



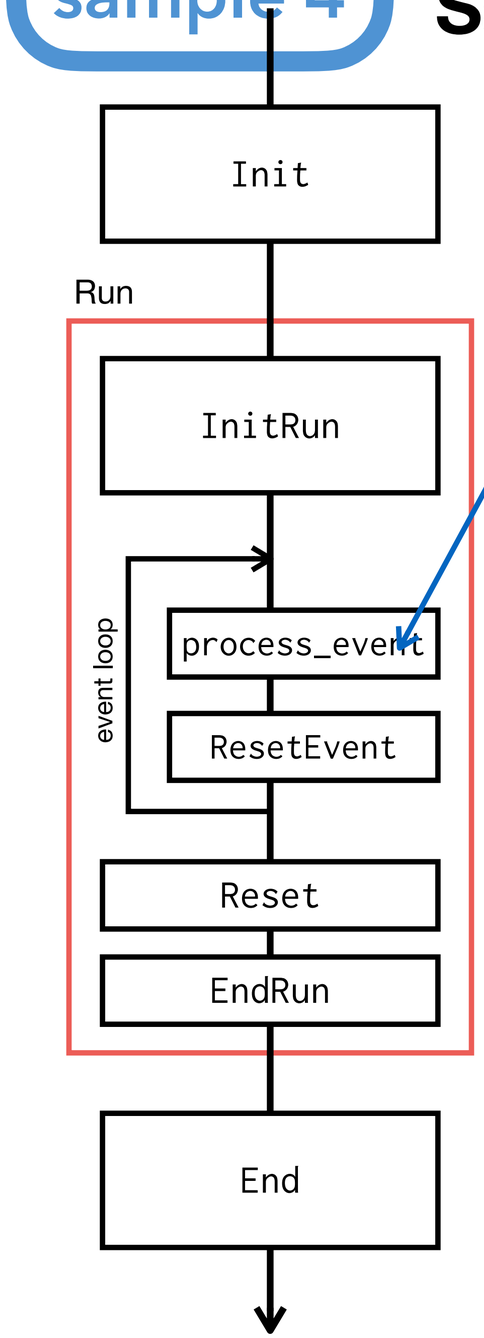
Rest part of tutorial::cluster_analysis

All get functions of TrkrCluster are written here

```
            // Then, reset the parameters (it's not mandatory if all parameters are available all the time. It's just in case)
            this->ResetClusterLoop();

            // If user wants to see cluster information, do it
            if( this->Verbosity() > 1 )
              {
                // All Get functions of TrkrCluster are here though some are commented out
                std::cout
                  << std::setw(6) << std::setprecision(3) << cluster->getPosition(0) << " " // ; // float
                  << std::setw(6) << std::setprecision(3) << cluster->getPosition(1) << " " // ; // float
                  << std::setw(6) << std::setprecision(3) << cluster->getPosition(2) << " " // ; // float
                  << std::setw(5) << std::setprecision(5) << cluster->getAdc() << " " // ; // unsigned int
                  // << std::setw(5) << std::setprecision(5) << cluster->getMaxAdc() << " " // ; // unsigned int

                  //<< std::setw(5) << std::setprecision(5) << cluster->getSize() << " " // ; // char, phi size * z size is returned
                  << std::setw(3) << std::setprecision(3) << cluster->getPhiSize() << " " // ; // float
                  //<< std::setw(2) << std::setprecision(2) << cluster->getZSize() << " " // ; // float
                  << std::endl;
                // cluster->getPosition( 0 ) ; // float , argument can be 0 or 1, not so useful
                // cluster->getLocalX() ; // float
                // cluster->getLocalY() ; // float
                // cluster->getSubSurfKey() ; // TrkrDefs::subsurfkey
                // cluster->getOverlap() ; // char
                // cluster->getEdge() ; // char

              } // End of if( this->Verbosity() > 1 )
          } // End of for (auto clusIter = range.first; clusIter != range.second; ++clusIter)
      } // End of for (const auto &hitsetkey : node_cluster_map->getHitSetKeys(TrkrDefs::TrkrId::inttId, inttlayer + 3) )
  } // End of for (unsigned int inttlayer = 0; inttlayer < 4; inttlayer++)

  if( this->Verbosity() > 0 )
    {
      std::cout << "Cluster num: " << std::setw(5) << cluster_num_ << std::endl;
    }

  return Fun4AllReturnCodes::EVENT_OK;
}
```

sample 4

Sample4: Analyzing TrkrCluster: sample_module_4/tutorial.cc



ResetClusterLoop function to be executed at the end of cluster loop

```
int tutorial::ResetClusterLoop()
{
  // Parameters to be resetted in the cluster loop are resettted

  // init/reset variables
  position_[0] = position_[1] = position_[2]
    = phi_ = theta_ = eta_
    = -9999.9;
  adc_ = size_phi_ = 0;

  return Fun4AllReturnCodes::EVENT_OK;
}
```

ResetEvent function to be executed at the end of event

```
int tutorial::ResetEvent(PHCompositeNode *topNode)
{

  this->ResetClusterLoop(); // Resetting cluster parameters

  run_num_ = event_id_
    = cluster_num_
    = cluster_num_layer_[ 0 ] = cluster_num_layer_[ 1 ]
    = cluster_num_layer_[ 2 ] = cluster_num_layer_[ 3 ]
    = 0;
  return Fun4AllReturnCodes::EVENT_OK;
}
```

sample 4

Analysis module #4: Hands on

HANDS ON! #8

Let's try

1. Compile and install sample_module_4

```
1.1. $ mv sample_module_4
1.2. $ mkdir build
1.3. $ mkdir install
1.4. $ cd build
1.5. $ ../autogen.sh --prefix=$PWD/../install
1.6. $ make install
```

2. Modify your environmental variable
   2.1. ROOT_INCLUDE_PATH
   2.2. LD_LIBRARY_PATH

After step 2.2, execute

```
$ export ROOT_INCLUDE_PATH=/sphenix/tg/tg01/commissioning/INTT/repositories/macros/common:${ROOT_INCLUDE_PATH}
```

Note: setup_local.sh always overwrite configurations.

3. Run Fun4All_minimum_4.C

```
3.1. $ root -q -b  'Fun4All_minimum_4.C( 1000 )'
```

Note: The official DST contains only 100 events. To analyze more events in a single process, you need to make a list of DSTs to be analyzed and use

```
Fun4AllInputManager::AddListFile
```

```
Fun4AllInputManager *in = new Fun4AllDstInputManager("DST");
in->AddListFile( list );
```

sample 4

Analysis module #4: Hands on

Let's try

4. Check the output ROOT file

```
[nukazuka@sphnx06 13:54:57 work_now] $ root tutorial_sample4.root
root [0]
Attaching file tutorial_sample4.root as _file0...
(TFile *) 0x1b39cc0
root [1] .ls
TFile**         tutorial_sample4.root
 TFile*         tutorial_sample4.root
  KEY: TTree    tree_event;1    Event base TTree
  KEY: TTree    tree_cluster;1  Cluster base TTree
root [2] tree_event->Print()
******************************************************************************
*Tree    :tree_event: Event base TTree                                       *
*Entries :      100 : Total =            5918 bytes  File  Size =       1736 *
*        :          : Tree compression factor =   1.00                       *
******************************************************************************
*Br    0 :run       : run_num/I                                              *
*Entries :      100 : Total  Size=       1057 bytes  One basket in memory    *
*Baskets :        0 : Basket Size=      32000 bytes  Compression=   1.00     *
*............................................................................*
*Br    1 :event     : event_id/I                                             *
*Entries :      100 : Total  Size=       1066 bytes  One basket in memory    *
*Baskets :        0 : Basket Size=      32000 bytes  Compression=   1.00     *
*............................................................................*
*Br    2 :cluster_num : cluster_num/I                                        *
*Entries :      100 : Total  Size=       1093 bytes  One basket in memory    *
*Baskets :        0 : Basket Size=      32000 bytes  Compression=   1.00     *
*............................................................................*
*Br    3 :cluster_num_layer : cluster_num_layer_[4]/I                        *
*Entries :      100 : Total  Size=       2338 bytes  One basket in memory    *
*Baskets :        0 : Basket Size=      32000 bytes  Compression=   1.00     *
*............................................................................*
```

```
root [3] tree_event->Scan()
******************************************************************************
*    Row   * Instance * run.run.r * event.eve * cluster_n * cluster_n *
******************************************************************************
*        0 *        0 *     51100 *        2 *       143 *        33 *
*        0 *        1 *     51100 *        2 *       143 *        32 *
*        0 *        2 *     51100 *        2 *       143 *        38 *
*        0 *        3 *     51100 *        2 *       143 *        40 *
*        1 *        0 *     51100 *        3 *       157 *        34 *
*        1 *        1 *     51100 *        3 *       157 *        39 *
*        1 *        2 *     51100 *        3 *       157 *        43 *
*        1 *        3 *     51100 *        3 *       157 *        41 *
*        2 *        0 *     51100 *        4 *       158 *        38 *
*        2 *        1 *     51100 *        4 *       158 *        37 *
*        2 *        2 *     51100 *        4 *       158 *        41 *
*        2 *        3 *     51100 *        4 *       158 *        42 *
*        3 *        0 *     51100 *        5 *       126 *        36 *
*        3 *        1 *     51100 *        5 *       126 *        26 *
*        3 *        2 *     51100 *        5 *       126 *        33 *
*        3 *        3 *     51100 *        5 *       126 *        31 *
*        4 *        0 *     51100 *        6 *       148 *        39 *
*        4 *        1 *     51100 *        6 *       148 *        29 *
*        4 *        2 *     51100 *        6 *       148 *        41 *
*        4 *        3 *     51100 *        6 *       148 *        39 *
*        5 *        0 *     51100 *        7 *       132 *        36 *
*        5 *        1 *     51100 *        7 *       132 *        30 *
*        5 *        2 *     51100 *        7 *       132 *        34 *
*        5 *        3 *     51100 *        7 *       132 *        32 *
*        6 *        0 *     51100 *        8 *        93 *        21 *
Type <CR> to continue or q to quit ==> q
******************************************************************************
(long long) 25
```

```
root [4] tree_cluster->Print()
******************************************************************************
*Tree    :tree_cluster: Cluster base TTree
*Entries :    23399 : Total =        1037917 bytes  File  Size =     259813 *
*        :          : Tree compression factor =   3.46                       *
******************************************************************************
*Br    0 :run       : run_num/I                                              *
*Entries :    23399 : Total  Size=      94421 bytes  File Size  =        542 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression= 118.07     *
*............................................................................*
*Br    1 :event     : event_id/I                                             *
*Entries :    23399 : Total  Size=      94434 bytes  File Size  =        841 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=  76.09     *
*............................................................................*
*Br    2 :position  : position[3]/F                                          *
*Entries :    23399 : Total  Size=     282169 bytes  File Size  =     187717 *
*Baskets :        8 : Basket Size=      32000 bytes  Compression=   1.36     *
*............................................................................*
*Br    3 :layer     : layer/I                                                *
*Entries :    23399 : Total  Size=      94425 bytes  File Size  =        504 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression= 126.97     *
*............................................................................*
*Br    4 :adc       : adc/F                                                  *
*Entries :    23399 : Total  Size=      94409 bytes  File Size  =      20311 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   3.15     *
*............................................................................*
*Br    5 :size_phi  : size_phi_/F                                            *
*Entries :    23399 : Total  Size=      94452 bytes  File Size  =      10687 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression=   5.99     *
*............................................................................*
*Br    6 :phi       : phi/F                                                  *
*Entries :    23399 : Total  Size=      94400 bytes  File Size  =        546 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression= 117.21     *
*............................................................................*
*Br    7 :theta     : theta/F                                                *
*Entries :    23399 : Total  Size=      94425 bytes  File Size  =        550 *
*Baskets :        2 : Basket Size=      32000 bytes  Compression= 116.35     *
*............................................................................*
*Br    8 :eta       : eta/F                                                  *
*Entries :    23399 : Total  Size=      94400 bytes  File Size  =        546 *
```

```
root [5] tree_cluster->Scan()
******************************************************************************************************************
*    Row   * Instance * run.run.r * event.eve * position. * layer.lay *   adc.adc * size_phi. *   phi.phi * theta.the *
******************************************************************************************************************
*        0 *        0 *     51100 *        2 * 7.4983215 *        0 *       105 *        2 * -9999.900 * -9999.900 *
*        0 *        1 *     51100 *        2 * -1.451081 *        0 *       105 *        2 * -9999.900 * -9999.900 *
*        0 *        2 *     51100 *        2 * -1.372450 *        0 *       105 *        2 * -9999.900 * -9999.900 *
*        1 *        0 *     51100 *        2 * 7.1972456 *        0 *        90 *        1 * -9999.900 * -9999.900 *
*        1 *        1 *     51100 *        2 * 2.0042343 *        0 *        90 *        1 * -9999.900 * -9999.900 *
*        1 *        2 *     51100 *        2 * -9.372450 *        0 *        90 *        1 * -9999.900 * -9999.900 *
*        2 *        0 *     51100 *        2 * 7.1055202 *        0 *        45 *        1 * -9999.900 * -9999.900 *
*        2 *        1 *     51100 *        2 * 2.3430373 *        0 *        45 *        1 * -9999.900 * -9999.900 *
*        2 *        2 *     51100 *        2 * -2.972450 *        0 *        45 *        1 * -9999.900 * -9999.900 *
*        3 *        0 *     51100 *        2 * 5.9337325 *        0 *        90 *        1 * -9999.900 * -9999.900 *
*        3 *        1 *     51100 *        2 * 4.3771534 *        0 *        90 *        1 * -9999.900 * -9999.900 *
*        3 *        2 *     51100 *        2 * -4.572450 *        0 *        90 *        1 * -9999.900 * -9999.900 *
*        4 *        0 *     51100 *        2 * 5.5393333 *        0 *        60 *        2 * -9999.900 * -9999.900 *
*        4 *        1 *     51100 *        2 * 4.7714610 *        0 *        60 *        2 * -9999.900 * -9999.900 *
*        4 *        2 *     51100 *        2 * -1.372450 *        0 *        60 *        2 * -9999.900 * -9999.900 *
*        5 *        0 *     51100 *        2 * -2.467580 *        0 *        75 *        2 * -9999.900 * -9999.900 *
*        5 *        1 *     51100 *        2 * 6.6639084 *        0 *        75 *        2 * -9999.900 * -9999.900 *
*        5 *        2 *     51100 *        2 * -9.372450 *        0 *        75 *        2 * -9999.900 * -9999.900 *
*        6 *        0 *     51100 *        2 * -1.354925 *        0 *       570 *        4 * -9999.900 * -9999.900 *
*        6 *        1 *     51100 *        2 * 6.9715404 *        0 *       570 *        4 * -9999.900 * -9999.900 *
*        6 *        2 *     51100 *        2 * -7.772450 *        0 *       570 *        4 * -9999.900 * -9999.900 *
*        7 *        0 *     51100 *        2 * -1.704509 *        0 *        60 *        1 * -9999.900 * -9999.900 *
*        7 *        1 *     51100 *        2 * 6.8748860 *        0 *        60 *        1 * -9999.900 * -9999.900 *
*        7 *        2 *     51100 *        2 * -6.172450 *        0 *        60 *        1 * -9999.900 * -9999.900 *
*        8 *        0 *     51100 *        2 * -2.290909 *        0 *       480 *        3 * -9999.900 * -9999.900 *
Type <CR> to continue or q to quit ==> .q
```

HANDS ON! #8

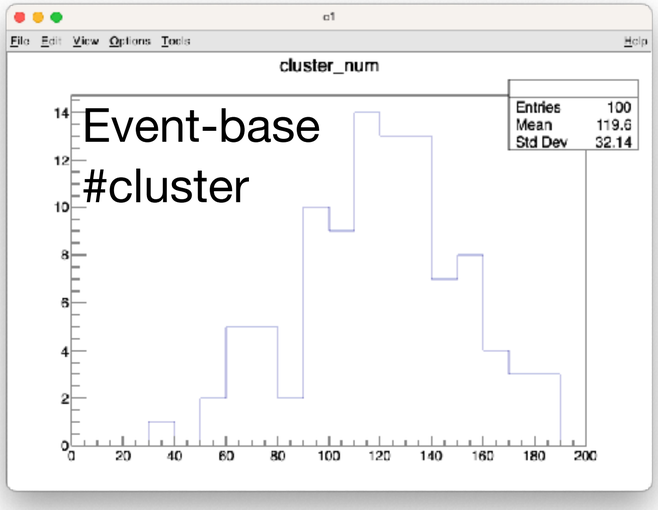
sample 4

Analysis module #4: Hands on

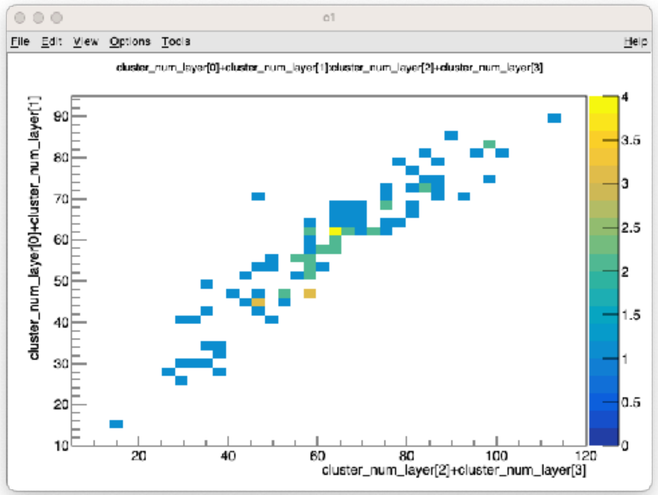
HANDS ON! #8

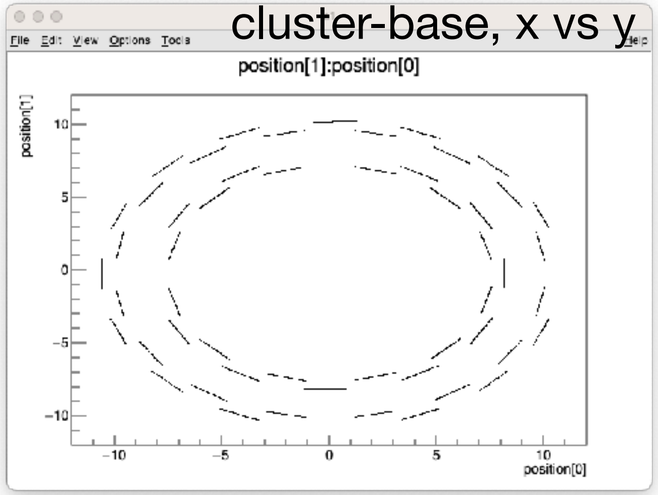
Let's try

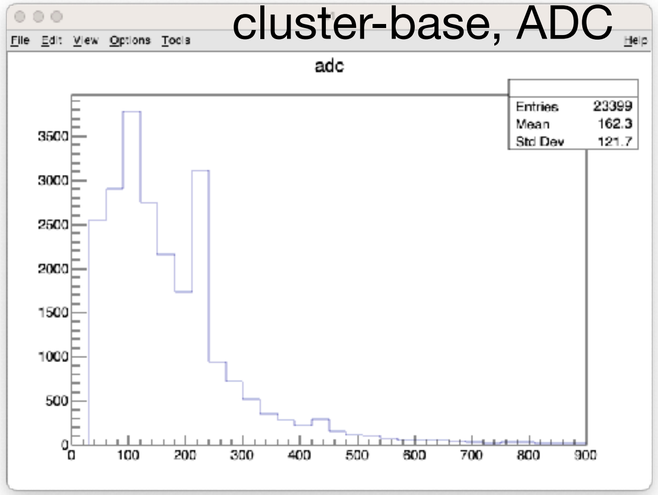
5. Draw some histograms

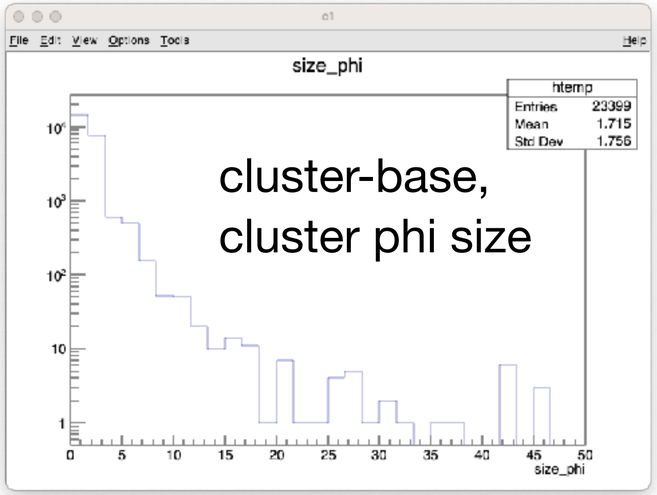


Event-base, #cluster on outer barrel vs inner barrel









sample 4

Analysis module #4: Hands on

Let's try

5. Draw some histograms (hint)
   5.1. NoMachine user: just run root command
   5.2. VS code user: look ROOT file in VS code
   5.3. terminal user: send ROOT file with scp command
   Example:
```
$ scp sphnx03:/sphenix/u/nukazuka/work_now/tutorial_sample4.root .
```

HANDS ON!
#8

sample 4

Analysis module #4: Homework

- Learn class inheritance in C++.
- Learn polymorphism.
- Learn the environment variable LD_LIBRARY_PATH
- Complete cluster ϕ, θ, η calculation. If some more information is needed for the calculation, just assume the simplest case.

```
153                 // Assign cluster parameters
154                 position_[0] = cluster->getPosition( 0 ); // x
155                 position_[1] = cluster->getPosition( 1 ); // y
156                 position_[2] = cluster->getPosition( 2 ); // z
157                 adc_ = cluster->getAdc();
158                 size_phi_ = cluster->getPhiSize();
159
160                 /** @TODO Calculate phi, theta, eta (pseudorapidity) by yourself
161                   phi_ = 0; // (radian)
162                   theta_ = 0; // (radian)
163                   eta = 0; // pseudorapidity
164                  */
```