Software Working Group: Introduction & Outlook

N. Baltzell

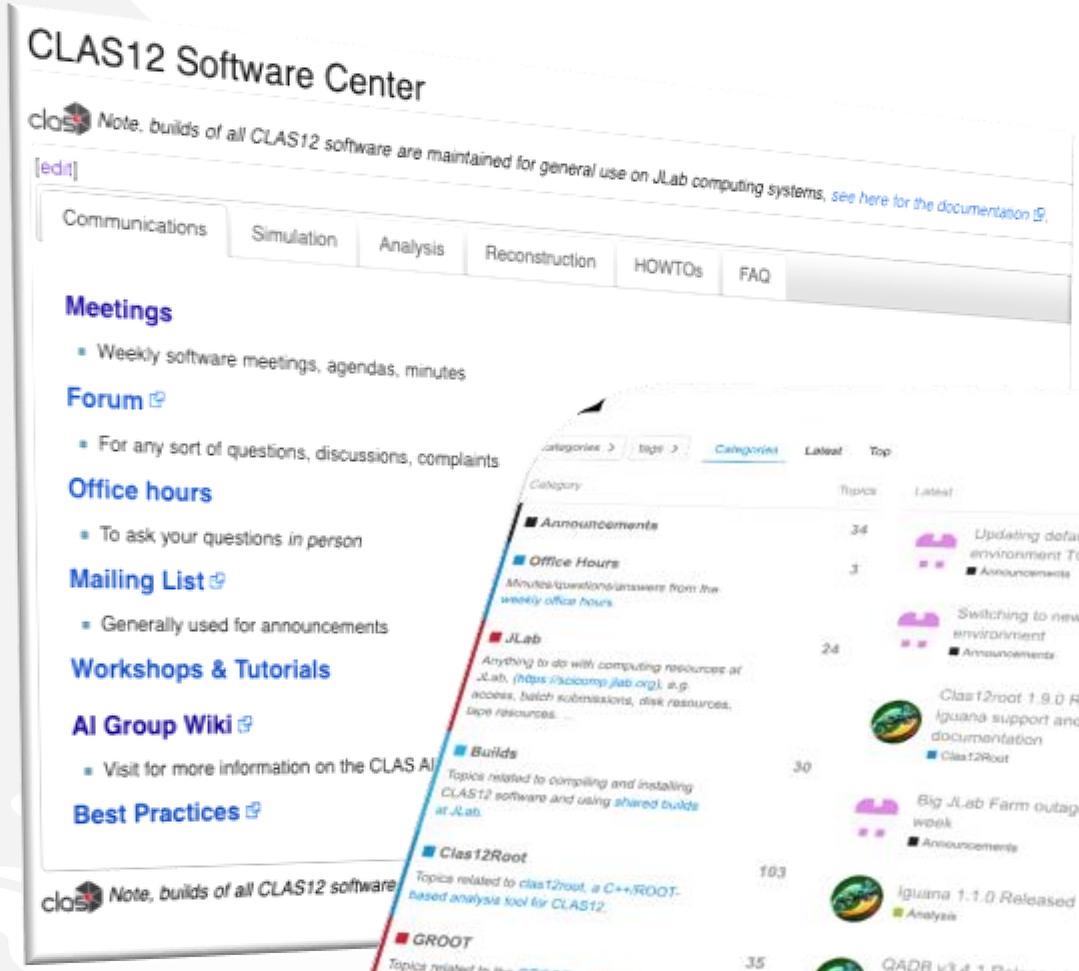
CLAS Collaboration Meeting

March 10, 2026

1. *CLAS software working group*
2. *Collaboration resources @ JLab*
3. *Computing landscape changes*
4. *Recent CLAS12 software progress and outlook*

CLAS12 Software Working Group

- We need 2 representatives for the Service Work committee, David Heddle and … you!
- SWWG chair election next year!
- Weekly meeting, Thursdays @ 11:00
  - last meeting of the month replaced by AI;)
- Office Hours, Tuesdays @ 9:30
- Software Wiki, Discourse Forum
- More tutorials, interactive demos, surveys



CLAS12 Software Center

Note, builds of all CLAS12 software are maintained for general use on JLab computing systems, see here for the documentation.

[edit]

Meetings
- Weekly software meetings, agendas, minutes

Forum
- For any sort of questions, discussions, complaints

Office hours
- To ask your questions *in person*

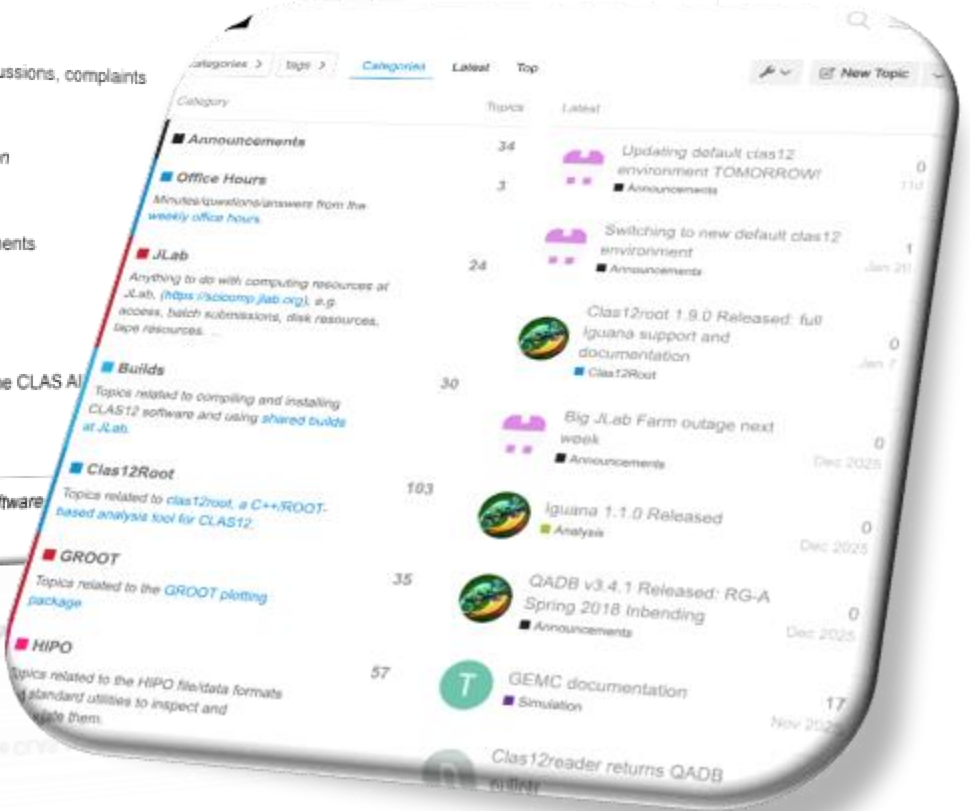
Mailing List
- Generally used for announcements

Workshops & Tutorials

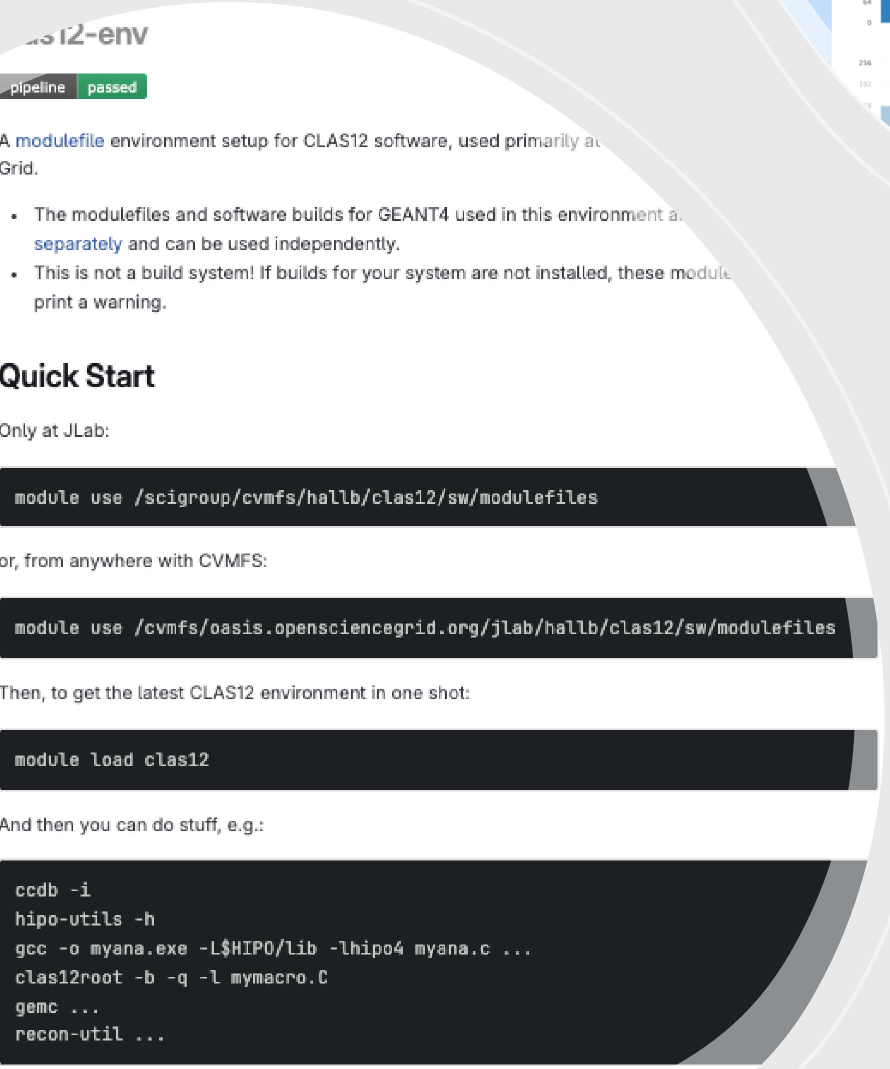
AI Group Wiki
- Visit for more information on the CLAS AI

Best Practices

Note, builds of all CLAS12 software



Category	Topics	Latest		
Announcements	34	Updating default clas12 environment TOMORROW! Announcements	0	11d
Office Hours: Minutes/questions/answers from the weekly office hours	3	Switching to new default clas12 environment Announcements	1	Jan 21
JLab: Anything to do with computing resources at JLab, (https://scicomp.jlab.org), e.g. access, batch submissions, disk resources, tape resources.	24	Clas12root 1.9.0 Released: full Iguana support and documentation Clas12Root	0	Jan 7
Builds: Topics related to compiling and installing CLAS12 software and using shared builds at JLab.	30	Big JLab Farm outage next week Announcements	0	Dec 2025
Clas12Root: Topics related to clas12root, a C++/ROOT-based analysis tool for CLAS12	103	Iguana 1.1.0 Released Analysis	0	Dec 2025
GROOT: Topics related to the GROOT plotting package	35	QADB v3.4.1 Released: RG-A Spring 2018 Inbending Announcements	0	Dec 2025
HIPO: Topics related to the HIPO file/data formats and standard utilities to inspect and [illegible] them.	57	GEMC documentation Simulation	17	Nov 2025
		Clas12reader returns QADB [illegible]		



s12-env

pipeline passed

A modulefile environment setup for CLAS12 software, used primarily a
Grid.

- The modulefiles and software builds for GEANT4 used in this environment a
  separately and can be used independently.
- This is not a build system! If builds for your system are not installed, these module
  print a warning.

Quick Start

Only at JLab:

```
module use /scigroup/cvmfs/hallb/clas12/sw/modulefiles
```

or, from anywhere with CVMFS:

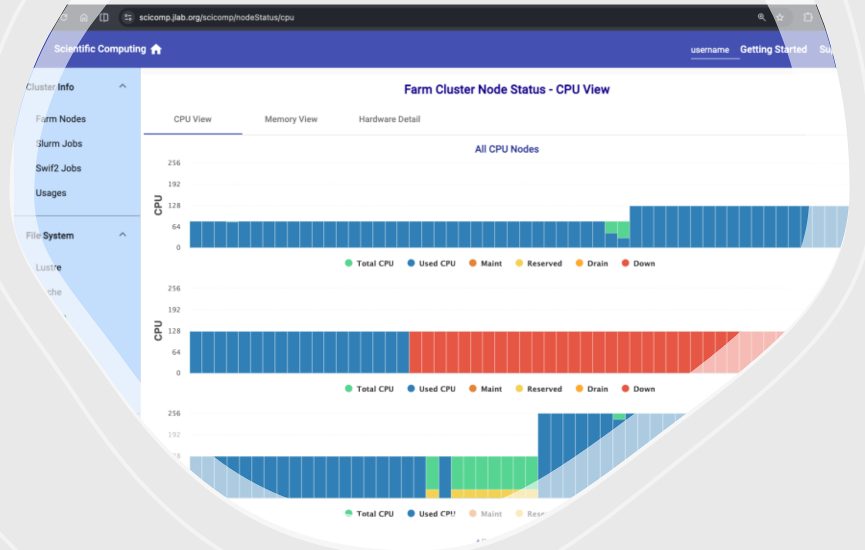
```
module use /cvmfs/oasis.opensciencegrid.org/jlab/hallb/clas12/sw/modulefiles
```

Then, to get the latest CLAS12 environment in one shot:

```
module load clas12
```

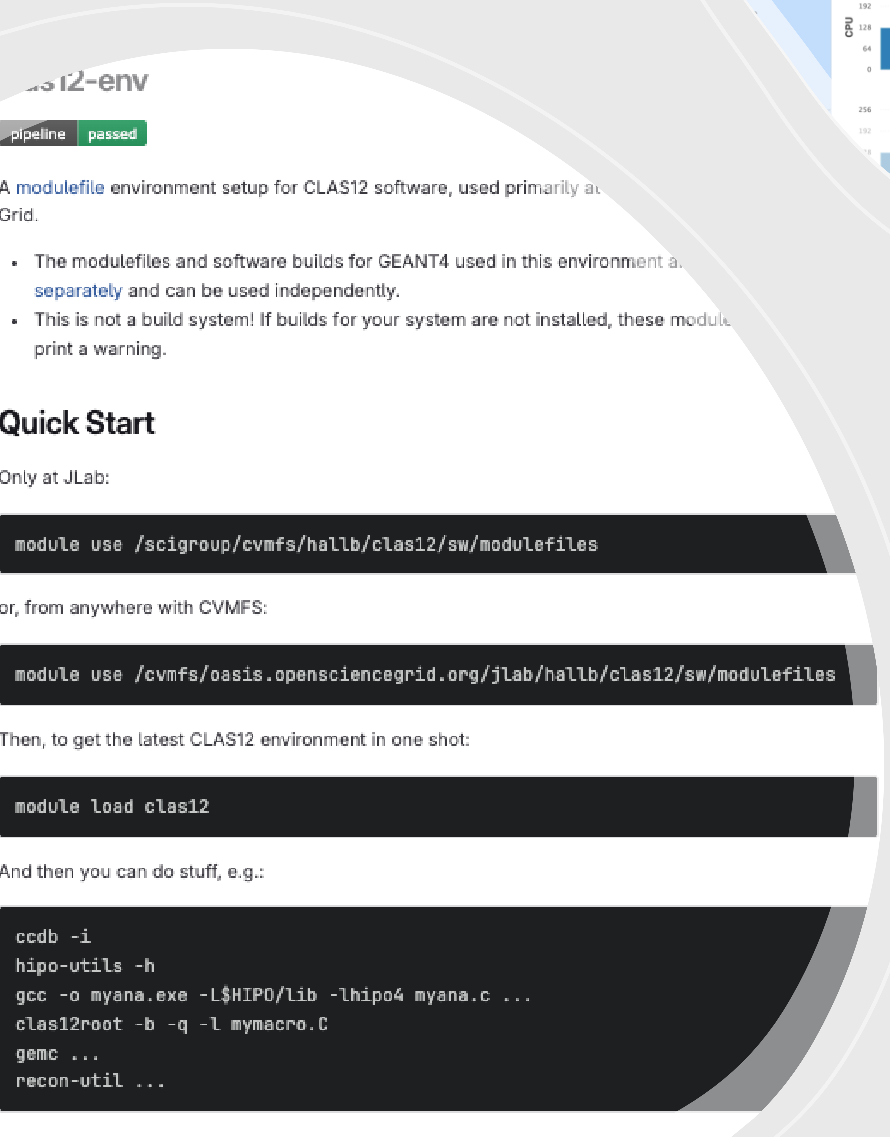
And then you can do stuff, e.g.:

```
ccdb -i
hipo-utils -h
gcc -o myana.exe -L$HIPO/lib -lhipo4 myana.c ...
clas12root -b -q -l mymacro.C
gemc ...
recon-util ...
```



Clas12 @ JLab: Getting started

- First, you'll need MFA, via a help-desk ticket at https://jlab.servicenowservices.com/
- Second, addition to the clas12 Unix/SLURM groups via baltzell@jlab.org
- Then "ssh ifarm" is round-robin access to nodes for interactive use
  - Note, a custom ssh configuration can minimize authentication headaches
- See scicomp.jlab.org for all general JLab computing documentation
- See clas12 environment modules for quick software access



s12-env

pipeline passed

A modulefile environment setup for CLAS12 software, used primarily a
Grid.

- The modulefiles and software builds for GEANT4 used in this environment a
  separately and can be used independently.
- This is not a build system! If builds for your system are not installed, these module
  print a warning.

Quick Start

Only at JLab:

```
module use /scigroup/cvmfs/hallb/clas12/sw/modulefiles
```

or, from anywhere with CVMFS:

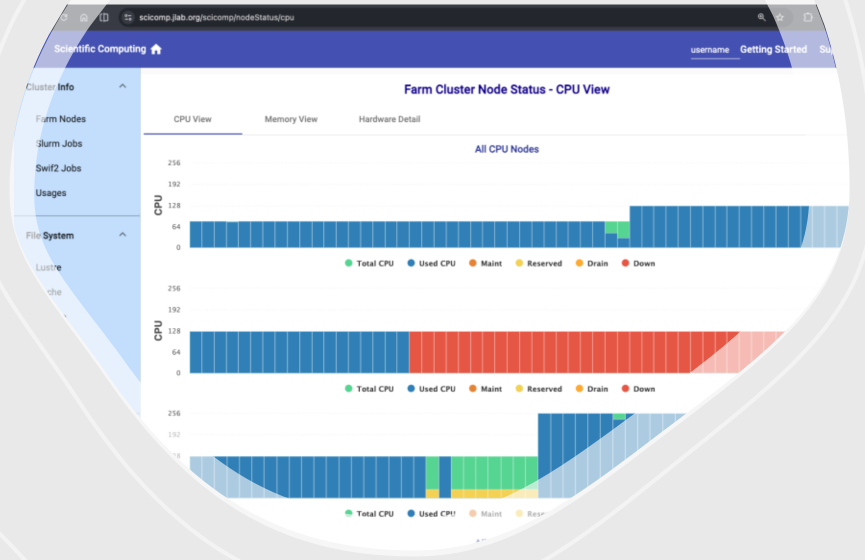
```
module use /cvmfs/oasis.opensciencegrid.org/jlab/hallb/clas12/sw/modulefiles
```

Then, to get the latest CLAS12 environment in one shot:

```
module load clas12
```

And then you can do stuff, e.g.:

```
ccdb -i
hipo-utils -h
gcc -o myana.exe -L$HIPO/lib -lhipo4 myana.c ...
clas12root -b -q -l mymacro.C
gemc ...
recon-util ...
```



Clas12 @ JLab: Round 2

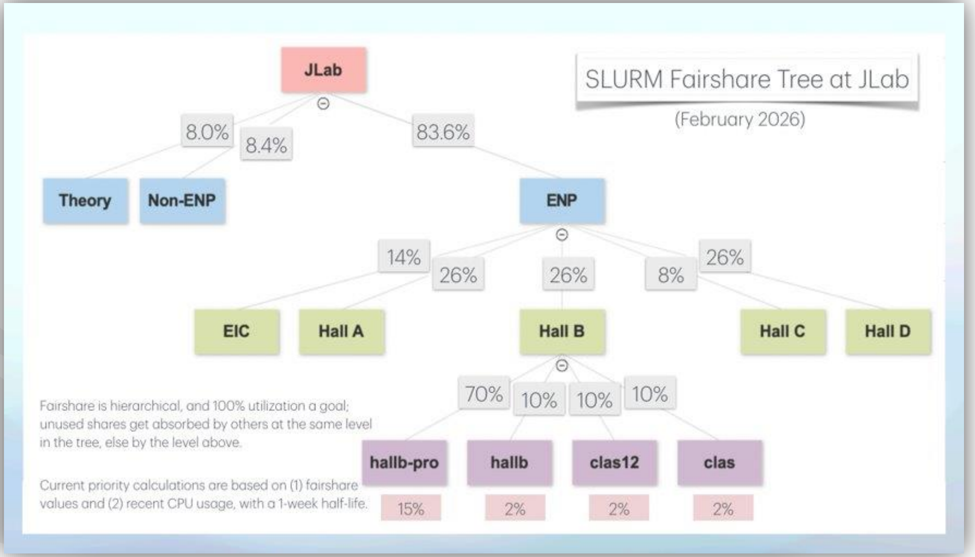
- /work, /volatile, /cache, /mss, /scratch filesystems
  - Note, /work/clas12b is a new addition in recent months
  - And, /volatile/clas12's lifetime is increasing significantly, as anticipated
- Analyzing large data sets often benefits from SWIF/SLURM
  - Note, farm job resource requests are important, ask for what you really need!
- JLab also provides dedicated JupyterHub resources
- JLab provides an in-house GitLab server and a paid GitHub organization for version control, CI, container/package registries, models/networks ...

JLab Agriculture State

- In 2024, ~10% of ENP compute was taken offline to power non-ENP hardware, due to infrastructure shortages. It has not returned.
- In early 2025, the farm25 nodes were delivered, a ~20% planned CPU increase. They're still offline. That's at least 10% of their usable lifetime and ~20% of warranty.
- In late 2025, another ~20% of ENP compute was taken offline due to infrastructure shortages. It has yet to return.

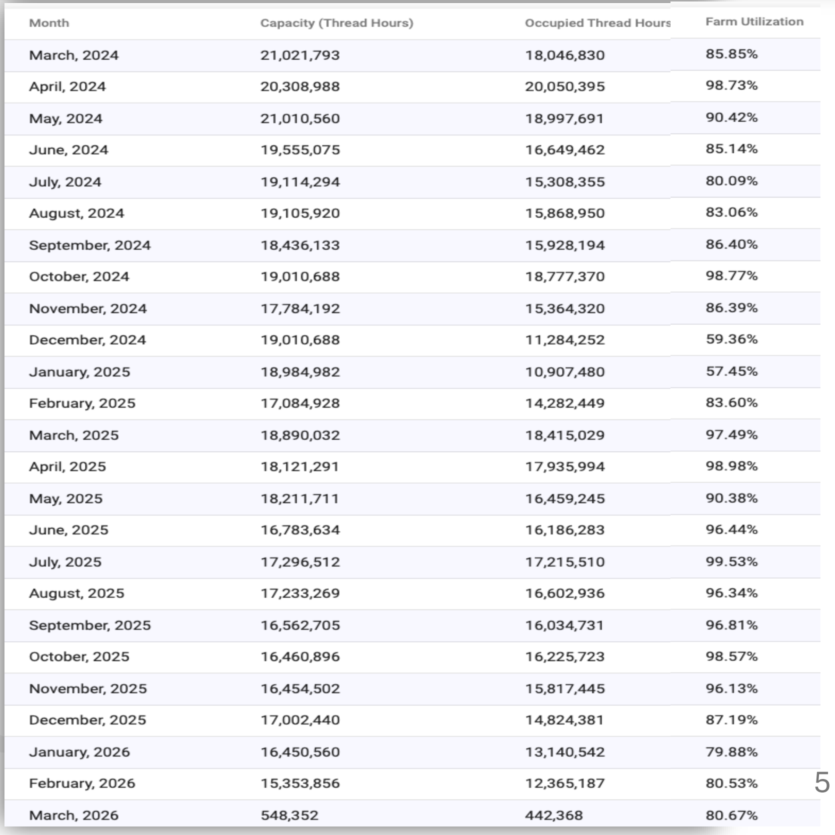
➞ *ENP/CEBAF compute is, and has been for a few months now, down ~2x from where it was planned to be only one year prior.*

➞ *Every year, the halls provide 5-year resource projections, cpu hours per year, and a 2026 purchase had been anticipated.*



Fairshare is hierarchical, and 100% utilization a goal; unused shares get absorbed by others at the same level in the tree, else by the level above.

Current priority calculations are based on (1) fairshare values and (2) recent CPU usage, with a 1-week half-life.



Month	Capacity (Thread Hours)	Occupied Thread Hours	Farm Utilization
March, 2024	21,021,793	18,046,830	85.85%
April, 2024	20,308,988	20,050,395	98.73%
May, 2024	21,010,560	18,997,691	90.42%
June, 2024	19,555,075	16,649,462	85.14%
July, 2024	19,114,294	15,308,355	80.09%
August, 2024	19,105,920	15,868,950	83.06%
September, 2024	18,436,133	15,928,194	86.40%
October, 2024	19,010,688	18,777,370	98.77%
November, 2024	17,784,192	15,364,320	86.39%
December, 2024	19,010,688	11,284,252	59.36%
January, 2025	18,984,982	10,907,480	57.45%
February, 2025	17,084,928	14,282,449	83.60%
March, 2025	18,890,032	18,415,029	97.49%
April, 2025	18,121,291	17,935,994	98.98%
May, 2025	18,211,711	16,459,245	90.38%
June, 2025	16,783,634	16,186,283	96.44%
July, 2025	17,296,512	17,215,510	99.53%
August, 2025	17,233,269	16,602,936	96.34%
September, 2025	16,562,705	16,034,731	96.81%
October, 2025	16,460,896	16,225,723	98.57%
November, 2025	16,454,502	15,817,445	96.13%
December, 2025	17,002,440	14,824,381	87.19%
January, 2026	16,450,560	13,140,542	79.88%
February, 2026	15,353,856	12,365,187	80.53%
March, 2026	548,352	442,368	80.67%

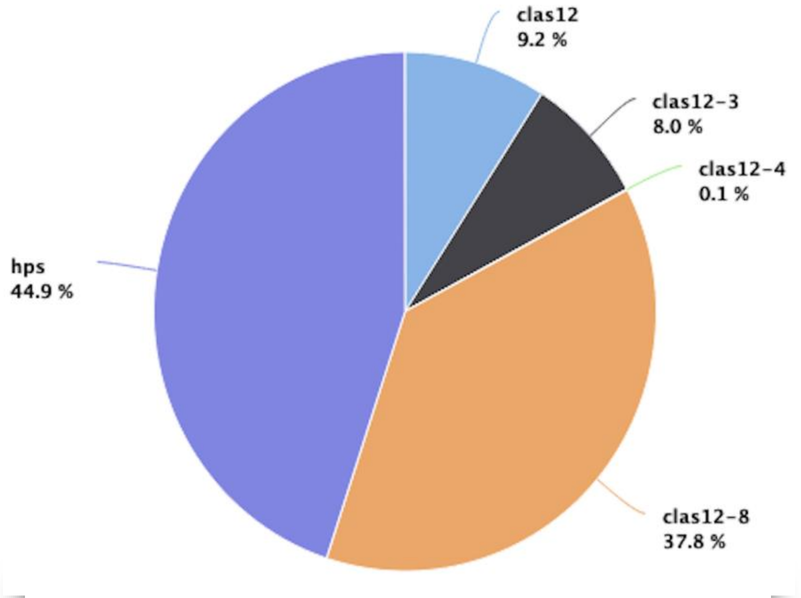
JLab Sharecropping

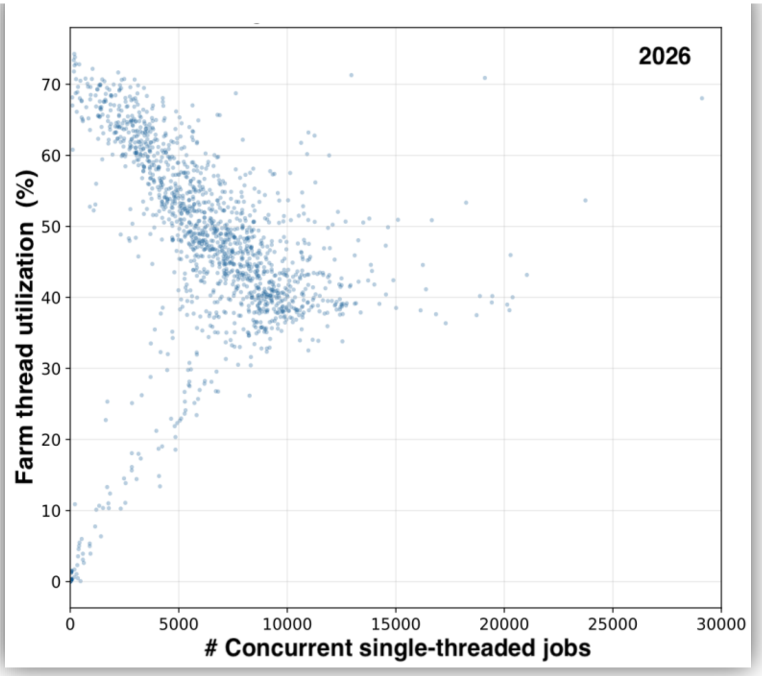
- In early 2025, as planned in the year prior, ENP's internal fairshare was redistributed, reducing B and D by one-third each to support an experiment in A
- Shortly later, the scheduling and accounting system was also modified to better support single-CPU jobs
  - A job slot at JLab had been a (hyper) thread, logical core, not a physical CPU
  - Previously, 1-slot, single-threaded jobs got paired with another, similar job on the same CPU, and (both) ran at half-speed
  - *Now, jobs requesting one slot get one physical CPU, and that CPU's other thread slot is reserved, for free*

→ These changes have knock-on effects

→ *Dynamic farm size for large jobs, adding uncertainty to projections.*

→ *An additional, hidden, relative fairshare redistribution between large and single-CPU jobs, e.g., ~25% for the 1:1:1, A:B:D split*





CLAS12 Software:

Recent Highlights and Outlook

Finish line ...

- RICH alignment (see Connor's talk today)
- Real-run-number simulations (see Raffaella's talk today)
- OSG portal fairshare upgrade (see Mauri's slides from last week)
- Clarafication, decoding speedups, reduced configuration overhead, increase workflow portability and reproducibility
- GEMC 5.13 / 5.12 release notes
- COATJAVA 13.7.1, 13.6.0, 13.5.3

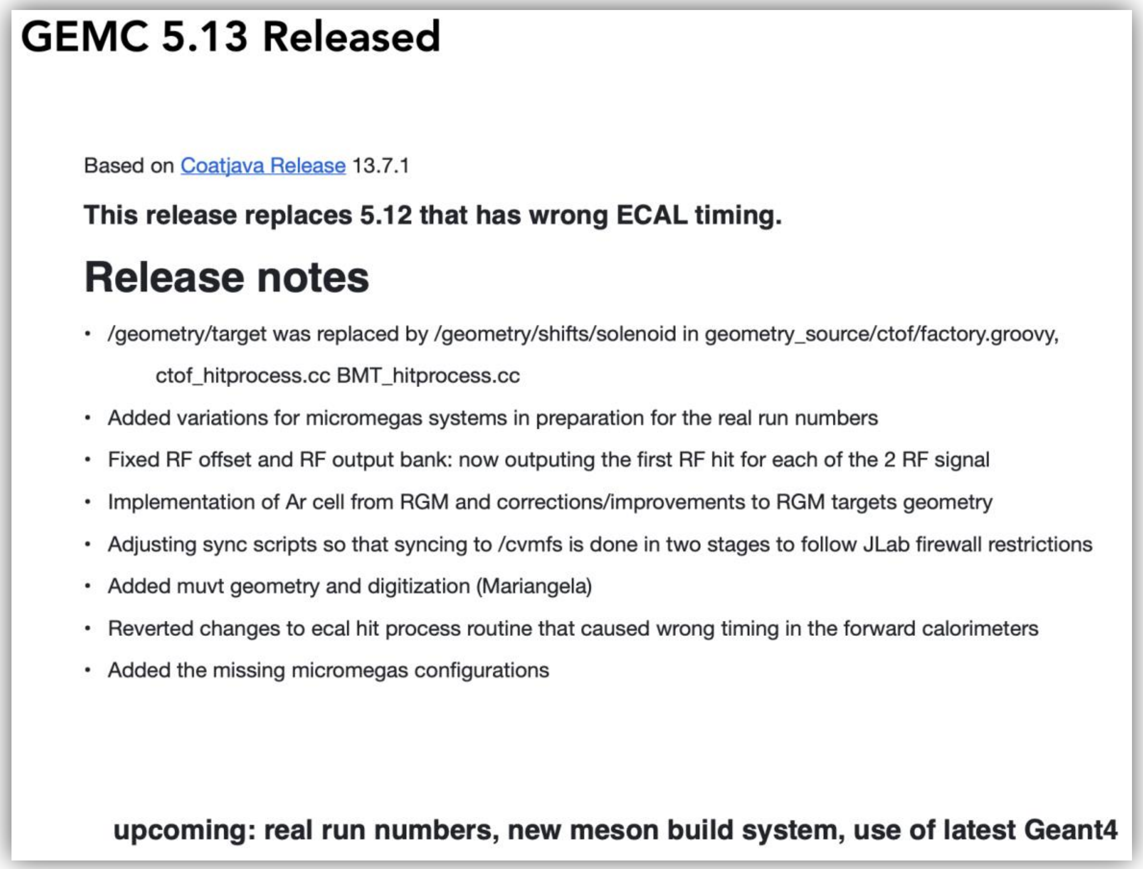
Last lap ...

- RG-L/ALERT detector AI advances (*see today's talks by Uditha and Mathieu*)
- DC AI (see Tongtong's talk today)
- Iguana, Inc., improved PID schemes, clas12root integration
- μRwell simulations (see Raffaella's talk at the μCLAS12 workshop)
- mcdj for usability, portability
- Profiling, benchmarking, scaling, CI
- calcode streamlining
- Speedups: swimming, tracking, CPU efficiency

Next ...

- Offsite resources (CNAF, NERSC, ...)
- Kinematic fitting (Iguana)
- Trigger simulation
- Etc.....

Updates: GEMC and OSG



GEMC 5.13 Released

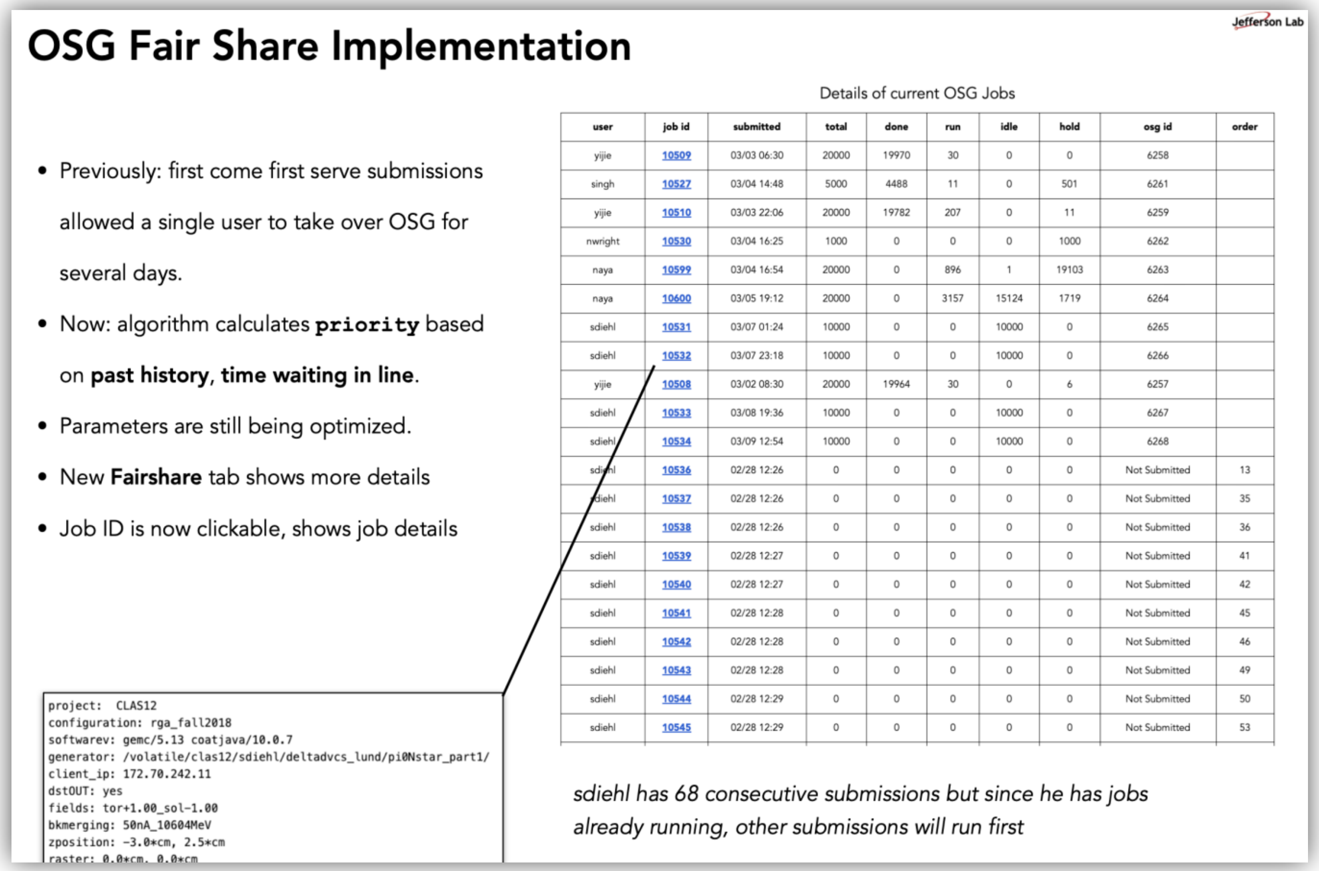
Based on Coatjava Release 13.7.1

This release replaces 5.12 that has wrong ECAL timing.

Release notes

- /geometry/target was replaced by /geometry/shifts/solenoid in geometry_source/ctof/factory.groovy, ctof_hitprocess.cc BMT_hitprocess.cc
- Added variations for micromegas systems in preparation for the real run numbers
- Fixed RF offset and RF output bank: now outputing the first RF hit for each of the 2 RF signal
- Implementation of Ar cell from RGM and corrections/improvements to RGM targets geometry
- Adjusting sync scripts so that syncing to /cvmfs is done in two stages to follow JLab firewall restrictions
- Added muvt geometry and digitization (Mariangela)
- Reverted changes to ecal hit process routine that caused wrong timing in the forward calorimeters
- Added the missing micromegas configurations

upcoming: real run numbers, new meson build system, use of latest Geant4



OSG Fair Share Implementation

Jefferson Lab

- Previously: first come first serve submissions allowed a single user to take over OSG for several days.
- Now: algorithm calculates `priority` based on **past history**, **time waiting in line**.
- Parameters are still being optimized.
- New **Fairshare** tab shows more details
- Job ID is now clickable, shows job details

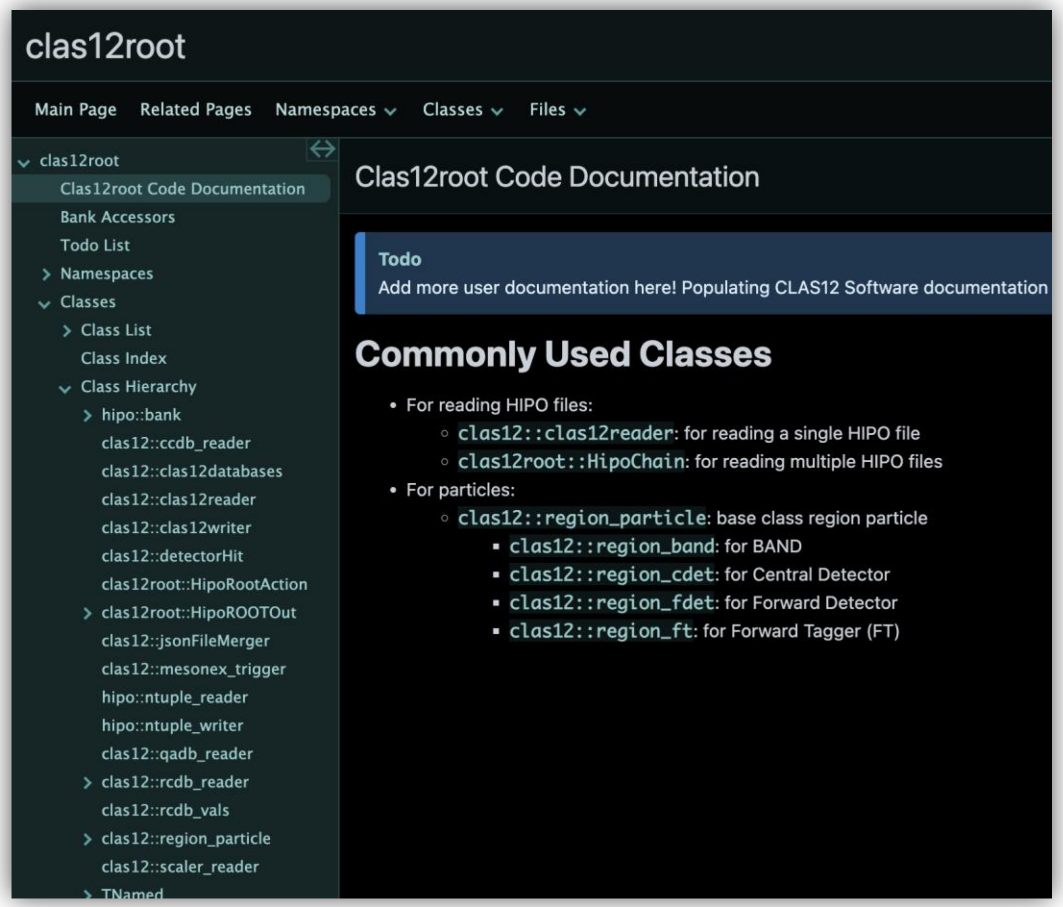
Details of current OSG Jobs

user	job id	submitted	total	done	run	idle	hold	osg id	order
yijie	10509	03/03 06:30	20000	19970	30	0	0	6258	
singh	10527	03/04 14:48	5000	4488	11	0	501	6261	
yijie	10510	03/03 22:06	20000	19782	207	0	11	6259	
nwright	10530	03/04 16:25	1000	0	0	0	1000	6262	
naya	10599	03/04 16:54	20000	0	896	1	19103	6263	
naya	10600	03/05 19:12	20000	0	3157	15124	1719	6264	
sdiehl	10531	03/07 01:24	10000	0	0	10000	0	6265	
sdiehl	10532	03/07 23:18	10000	0	0	10000	0	6266	
yijie	10508	03/02 08:30	20000	19964	30	0	6	6257	
sdiehl	10533	03/08 19:36	10000	0	0	10000	0	6267	
sdiehl	10534	03/09 12:54	10000	0	0	10000	0	6268	
sdiehl	10536	02/28 12:26	0	0	0	0	0	Not Submitted	13
sdiehl	10537	02/28 12:26	0	0	0	0	0	Not Submitted	35
sdiehl	10538	02/28 12:26	0	0	0	0	0	Not Submitted	36
sdiehl	10539	02/28 12:27	0	0	0	0	0	Not Submitted	41
sdiehl	10540	02/28 12:27	0	0	0	0	0	Not Submitted	42
sdiehl	10541	02/28 12:28	0	0	0	0	0	Not Submitted	45
sdiehl	10542	02/28 12:28	0	0	0	0	0	Not Submitted	46
sdiehl	10543	02/28 12:28	0	0	0	0	0	Not Submitted	49
sdiehl	10544	02/28 12:29	0	0	0	0	0	Not Submitted	50
sdiehl	10545	02/28 12:29	0	0	0	0	0	Not Submitted	53

```
project:  CLAS12
configuration: rga_fall2018
softwarev: gemc/5.13 coatjava/10.0.7
generator: /volatile/clas12/sdiehl/deltadvcs_lund/pi0Nstar_part1/
client_ip: 172.70.242.11
dstOUT: yes
fields: tor+1.00_sol-1.00
bkmerging: 50nA_10604MeV
zposition: -3.0*cm, 2.5*cm
raster: 0.0*cm, 0.0*cm
```

sdiehl has 68 consecutive submissions but since he has jobs already running, other submissions will run first

Updates: Analysis Software

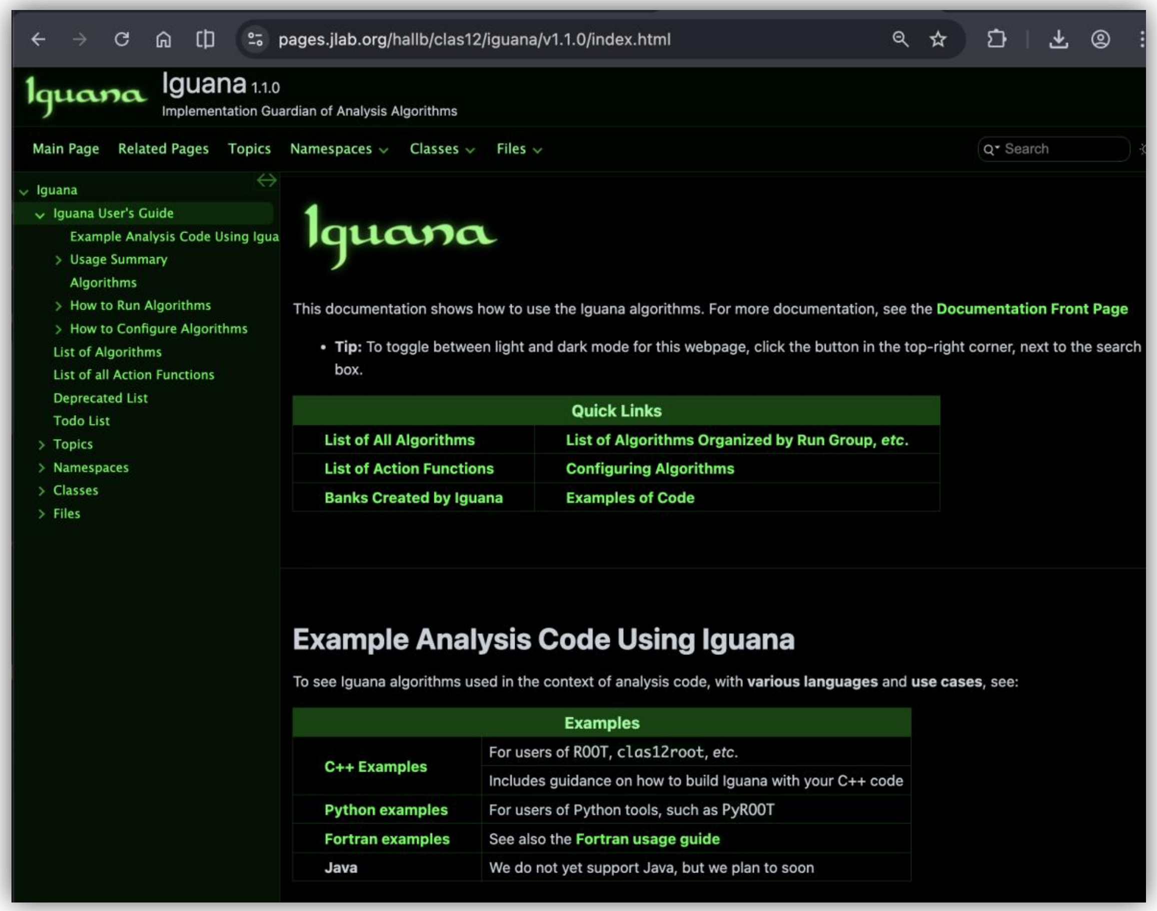


Clas12root Code Documentation

Todo
Add more user documentation here! Populating CLAS12 Software documentation

Commonly Used Classes

- For reading HIPO files:
  - `clas12::clas12reader`: for reading a single HIPO file
  - `clas12root::HipoChain`: for reading multiple HIPO files
- For particles:
  - `clas12::region_particle`: base class region particle
    - `clas12::region_band`: for BAND
    - `clas12::region_cdet`: for Central Detector
    - `clas12::region_fdet`: for Forward Detector
    - `clas12::region_ft`: for Forward Tagger (FT)



This documentation shows how to use the Iguana algorithms. For more documentation, see the **Documentation Front Page**

- **Tip:** To toggle between light and dark mode for this webpage, click the button in the top-right corner, next to the search box.

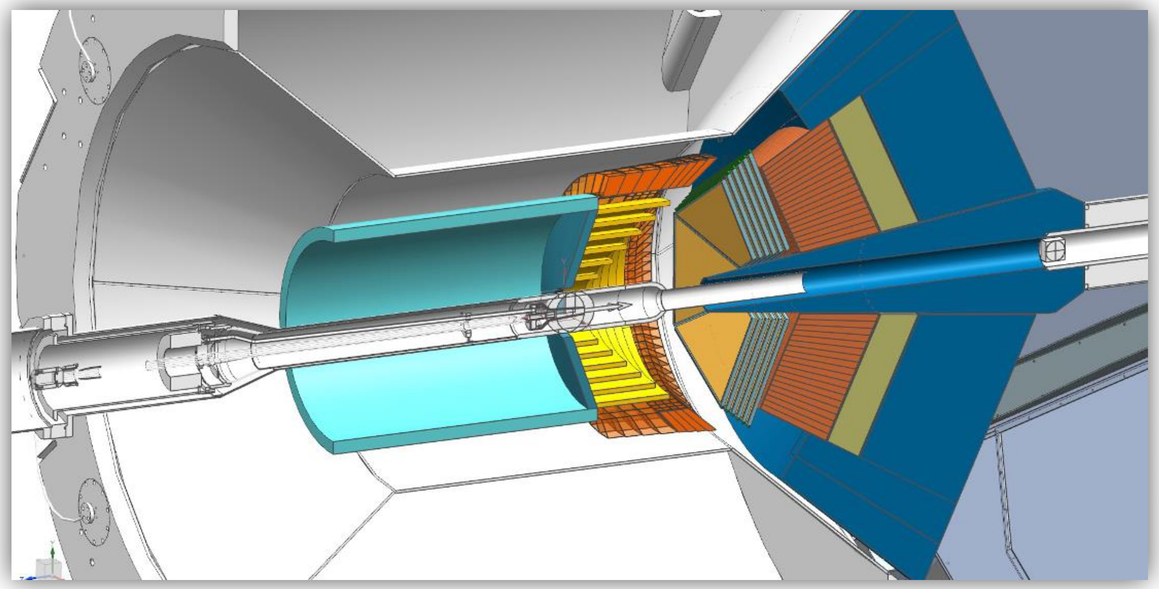
Quick Links	
List of All Algorithms	**List of Algorithms Organized by Run Group, *etc.***
List of Action Functions	**Configuring Algorithms**
Banks Created by Iguana	**Examples of Code**

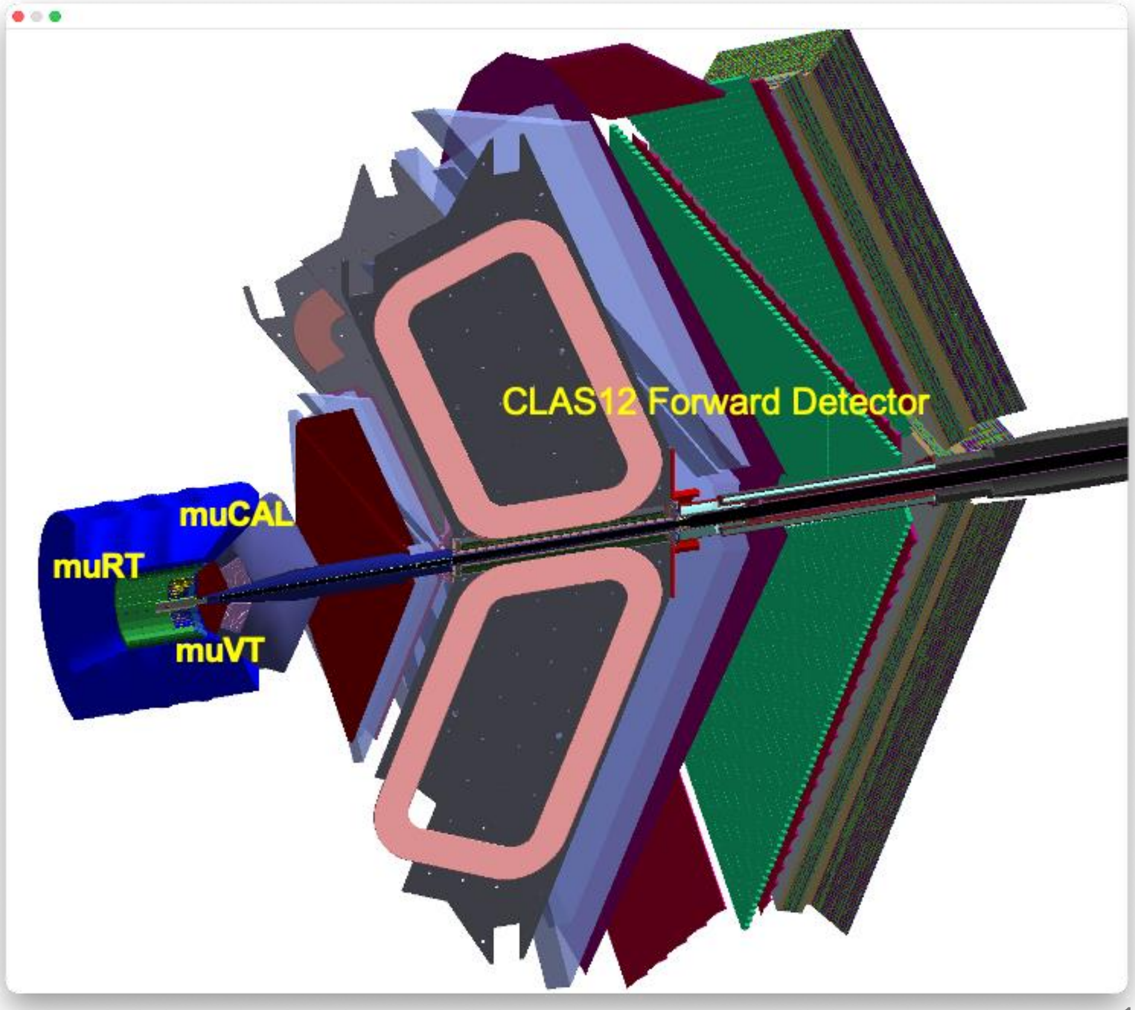
Example Analysis Code Using Iguana

To see Iguana algorithms used in the context of analysis code, with **various languages** and **use cases**, see:

Examples	
C++ Examples	For users of ROOT, `clas12root`, *etc.*
	Includes guidance on how to build Iguana with your C++ code
Python examples	For users of Python tools, such as PyROOT
Fortran examples	See also the **Fortran usage guide**
Java	We do not yet support Java, but we plan to soon

Update: μCLAS12 Simulations



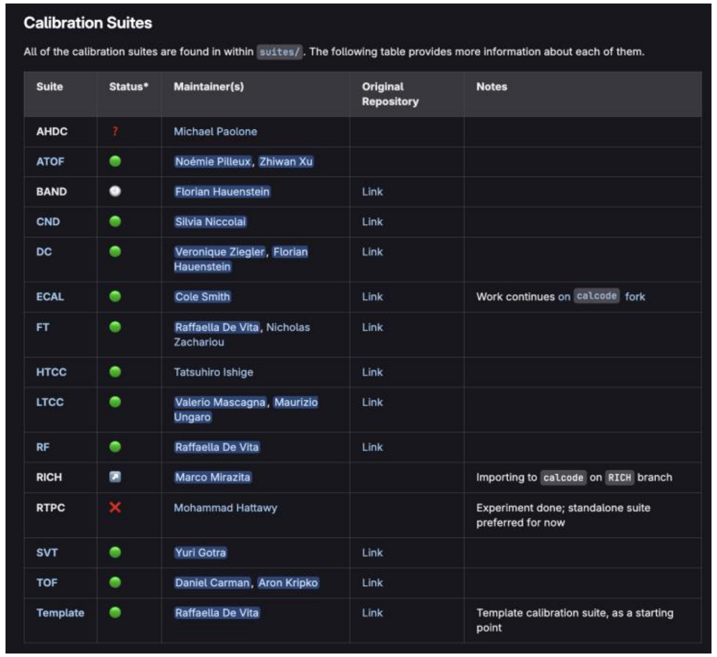


GEMC rendering of the muCLAS12 setup

Update: little things

- Standalone, unintegrated software for decoding, denoising, bg-merging, etc. makes for a complex workflow
- YAML (and GCARD) configuration complexity doesn't help, with absolute paths to standard schema directories, lots of CCDB variations, ever-increasing CLARA service chain lengths
- Calibration suites only support interactive, graphical mode, but need to read many GB of data in one shot for one calibration ➞ ssh connection reliance, manual data copying, unnecessary lag
- Running simulation jobs is complex, everyone needs their own "MCWrapper"

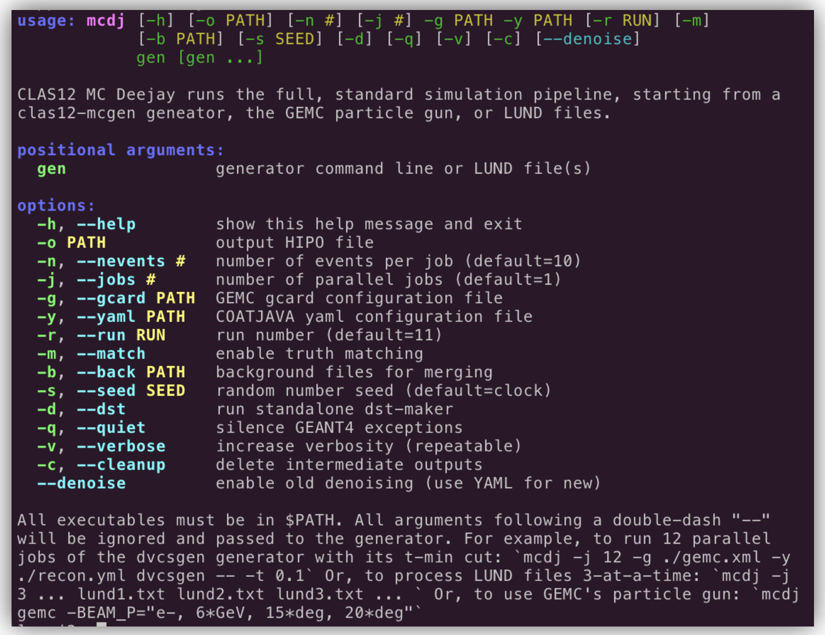
- All these are issues of portability, scalability, maintenance, etc ...
- ***Many have been addressed in the past 6 months, partially in preparation for offsite cooking, some still in progress, RG-L serving as first adopter for the cheffing ones (thanks Mathieu!), real run number simulations are related!***



Calibration Suites

All of the calibration suites are found in within `suites/`. The following table provides more information about each of them.

Suite	Status*	Maintainer(s)	Original Repository	Notes
AHDC	?	Michael Paolone		
ATOF	🟢	Noémie Pilleux, Zhiwan Xu		
BAND	⚪	Florian Hauenstein	Link	
CND	🟢	Silvia Niccolai	Link	
DC	🟢	Veronique Ziegler, Florian Hauenstein	Link	
ECAL	🟢	Cole Smith	Link	Work continues on `calcode` fork
FT	🟢	Raffaella De Vita, Nicholas Zachariou	Link	
HTCC	🟢	Tatsuhiro Ishige	Link	
LTCC	🟢	Valerio Mascagna, Maurizio Ungaro	Link	
RF	🟢	Raffaella De Vita	Link	
RICH	↗	Marco Mirazita		Importing to `calcode` on `RICH` branch
RTPC	❌	Mohammad Hattawy		Experiment done; standalone suite preferred for now
SVT	🟢	Yuri Gotra	Link	
TOF	🟢	Daniel Carman, Aron Kripko	Link	
Template	🟢	Raffaella De Vita	Link	Template calibration suite, as a starting point



```
usage: mcdj [-h] [-o PATH] [-n #] [-j #] -g PATH -y PATH [-r RUN] [-m]
            [-b PATH] [-s SEED] [-d] [-q] [-v] [-c] [--denoise]
            gen [gen ...]

CLAS12 MC Deejay runs the full, standard simulation pipeline, starting from a
clas12-mcgen geneator, the GEMC particle gun, or LUND files.

positional arguments:
  gen                   generator command line or LUND file(s)

options:
  -h, --help            show this help message and exit
  -o PATH               output HIPO file
  -n, --nevents #       number of events per job (default=10)
  -j, --jobs #          number of parallel jobs (default=1)
  -g, --gcard PATH      GEMC gcard configuration file
  -y, --yaml PATH       COATJAVA yaml configuration file
  -r, --run RUN         run number (default=11)
  -m, --match           enable truth matching
  -b, --back PATH       background files for merging
  -s, --seed SEED       random number seed (default=clock)
  -d, --dst             run standalone dst-maker
  -q, --quiet           silence GEANT4 exceptions
  -v, --verbose         increase verbosity (repeatable)
  -c, --cleanup         delete intermediate outputs
  --denoise             enable old denoising (use YAML for new)

All executables must be in $PATH. All arguments following a double-dash "--"
will be ignored and passed to the generator. For example, to run 12 parallel
jobs of the dvcsgen generator with its t-min cut: `mcdj -j 12 -g ./gemc.xml -y
./recon.yml dvcsgen -- -t 0.1` Or, to process LUND files 3-at-a-time: `mcdj -j
3 ... lund1.txt lund2.txt lund3.txt ... ` Or, to use GEMC's particle gun: `mcdj
gemc -BEAM_P="e-, 6*GeV, 15*deg, 20*deg"`
```

CLAS12 Software: Recent Highlights and Outlook

Finish line ...

- RICH alignment (see Connor's talk today)
- Real-run-number simulations (see Raffaella's talk today)
- OSG portal fairshare upgrade (see Mauri's slides from last week)
- Clarafication, decoding speedups, reduced configuration overhead, increase workflow portability and reproducibility
- GEMC 5.13 / 5.12 release notes
- COATJAVA 13.7.1, 13.6.0, 13.5.3

Last lap ...

- RG-L/ALERT detector AI advances (*see today's talks by Uditha and Mathieu*)
- DC AI (see Tongtong's talk today)
- Iguana, Inc., improved PID schemes, clas12root integration
- μRwell simulations (see Raffaella's talk at the μCLAS12 workshop)
- mcdj for usability, portability
- Profiling, benchmarking, scaling, CI
- calcode streamlining
- Speedups: swimming, tracking, CPU efficiency

Next ...

- Offsite resources (CNAF, NERSC, ...)
- Kinematic fitting (Iguana)
- Trigger simulation
- Etc.....

FINE

Examples of 2025's Accounting Change

- Suppose a farm with 30k total threads/slots to allocate, corresponding to 15k physical CPUs
  - And 2 fairshare groups, X and Y, with equal, 50:50 fairshare
  - X runs single-threaded jobs at T_x events per second (Hz) per CPU
  - Y runs multi-threaded jobs at T_y events per second (Hz) per CPU

- Then, let the fairshare system do its job with full queues:
  - Old accounting: X and Y are both billed for, and receive, 15k slots (threads), corresponding to 7.5k CPUs
    - Throughput: $X = 7.5 * T_x$ kHz and $Y = 7.5 * T_y$ kHz
  - New accounting: X and Y are both billed for 10k slots, but X receives 10k CPUs, and Y receives 5k CPUs
    - Throughput: $X = 10 * T_x$ kHz and $Y = 5 * T_y$ kHz
  - ***Effect: X gains 33% throughput, Y loses 33% throughput***

- Or, manually manage the fairshare by giving each group 100% of the farm for equal wall times:
  - ***Effect: throughputs are unchanged, but Y pays twice the bill as X for the same wall hours on the same hardware (the same 33% as above)***

- *Note, for 3 equal fairshare groups X:Y:Y, the 2-group, 33% redistribution factor above becomes 25%*

	Old		New	
	threads	CPUs	threads	CPUs
X	15	7.5	10	10
Y	15	7.5	10	5
MIA	0	0	10	0

	Old		New	
	threads	CPUs	threads	CPUs
X	10	5	7.5	7.5
Y	10	5	7.5	3.75
Y	10	5	7.5	3.75
MIA	0	0	7.5	0

Extras

- Fairshare for single/multi-threaded jobs on a CPU farm is an old problem
  1. Is your farm for single-threaded, floating-point arithmetic, number-crunching jobs?
  2. Or is it to maximize throughput for I/O heavy, large jobs?
  3. Or, if it's something in between, need a *well-defined, static metric*.

- The timeline plot for cpu/slot/thread hours used by each fairshare group is no longer available at scicomp.jlab.org 😦

- Hall B/D's fairshare fraction of the entire JLab farm was 45/45% in 2019, reduced to ~39% in 2020, and now 22% in 2025 *(the early changes were advertised with corresponding hardware increases that haven't lasted)* 😦

- The job wall-time limit was changed from 3 days to 24 hours, to mitigate fairshare loss to low-priority groups with unlimited queues and job durations, then increased back to 48 hours a few months later for Hall A.