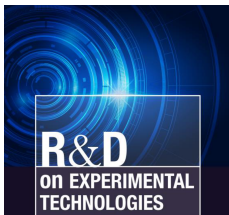


key4hep Update



Juan Miguel Carceller

on behalf of the Keyhep team



CERN

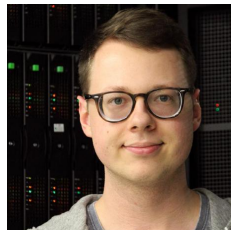
June 21, 2023

The key4hep Team: New Additions

- **Swathi Sasikumar** (fellow) joined in March



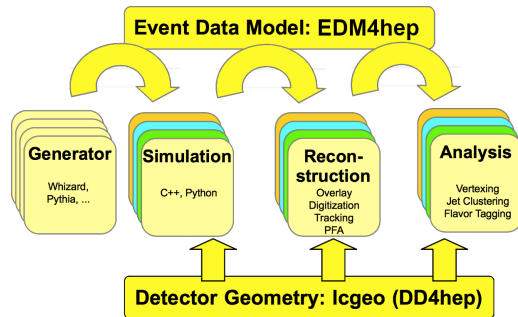
- **Leonhard Reichenbach** (PhD student) joined in March



- **Juan Miguel Carceller** (fellow) joined in February

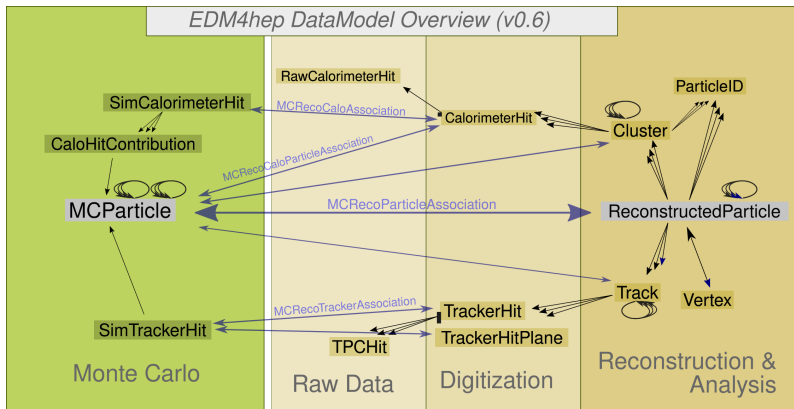


- Turnkey software for future accelerators
- Community with people from many different experiments: CEPC, CLIC, EIC, FCC, ILC, Muon Collider, etc.
- Share components to reduce maintenance and development cost and allow everyone to benefit from its improvements
- Complete data processing framework, from generation to data analysis



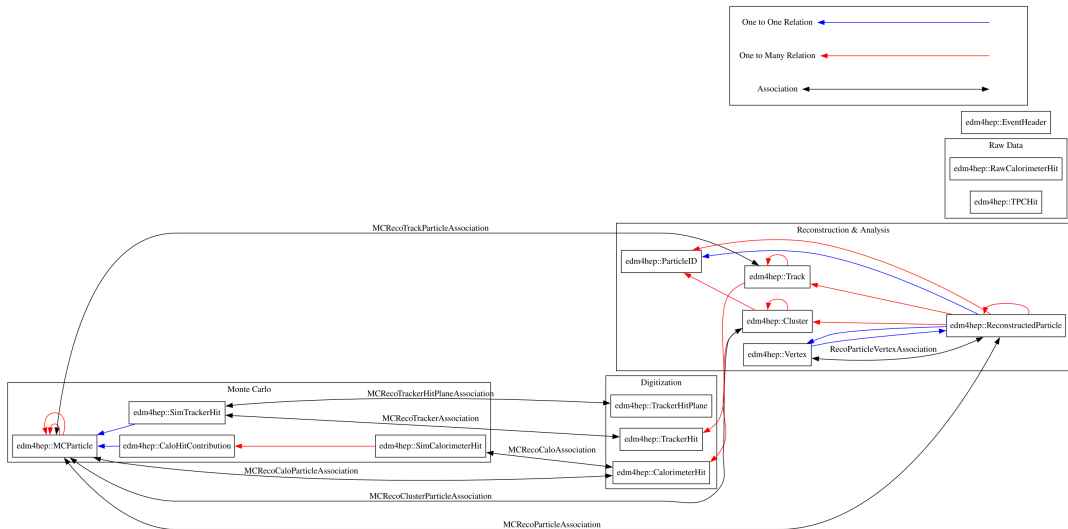
Event Data Model: EDM4hep

- Data Model used in key4hep
- From a specification in a yaml file, and using podio, the C++ code containing all the classes and methods is generated



Event Data Model: EDM4hep

- New tool to show a complete visualization of EDM4hep from the model description



Event Data Model: EDM4hep

- The data model itself is quite stable, only a few minor changes during the past months
- Changes on utilities around EDM4hep: visualization, python bindings, improved JSON support. . .
- There are now [python bindings](#) that make it possible to work with EDM4hep in python:

```
from edm4hep import edm4hep
particle = edm4hep.MCParticle() # default initialized particle
particle.getCharge() # 0.0

series = edm4hep.TimeSeries(1, 2, 3) # classes can be constructed with non-default parameters
series.getCellID() # 1
series.getTime() # 2.0
series.getInterval() # 3.0

mc = edm4hep.MutableMCParticle() # mutable classes can also be modified
mc.setPDG(2212)
mc.getPDG() # 2212
```

Event Data Model: EDM4hep

- Combined with the podio python bindings it's possible to write and read frames

```
from edm4hep import edm4hep
from ROOT import podio
from podio.root_io import Writer
import cppyy

writer = Writer('frame.root') # This will write the frame to a file called frame.root

frame = podio.Frame() # This is where the collection and other things will be stored
coll = edm4hep.MCParticleCollection() # Create an EDM4hep collection

coll.create() # Create a MutableMCParticle object

frame.put(cppyy.gbl.std.move(coll), "MCParticles") # Add collection to frame

writer.writeFrame(frame, 'events')
writer.finish()
```

- Yurii is already using these bindings to port scripts from the Muon Collider software to EDM4hep

Event Data Model: EDM4hep

- Combined with the podio python bindings it's possible to write and read frames

```
from edm4hep import edm4hep
from ROOT import podio
from podio.root_io import Writer
import cppyy

writer = Writer('frame.root') # This will write the frame to a file called frame.root

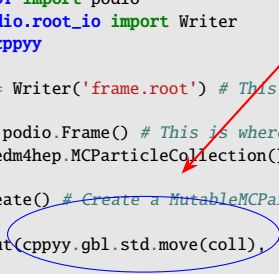
frame = podio.Frame() # This is where the collection and other things will be stored
coll = edm4hep.MCParticleCollection() # Create an EDM4hep collection

coll.create() # Create a MutableMCParticle object

frame.put(cppyy.gbl.std.move(coll), "MCParticles") # Add collection to frame

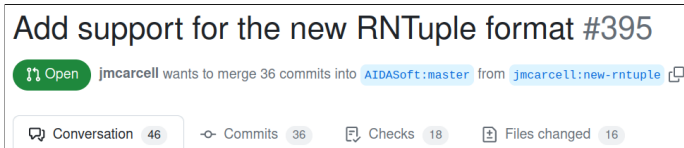
writer.writeFrame(frame, 'events')
writer.finish()
```

Ugly! Plans on improving it



- Yurii is already using these bindings to port scripts from the Muon Collider software to EDM4hep

- podio (Plain Old Data IO) is a toolkit for the creation of EDMs like EDM4hep
- Close to version 1.0: Schema evolution, among other improvements
- RNTuple backend: Long-standing PR, no major issues
 - Adds a RNTuple-based backend for writing and reading, using `podio::Frame`
 - Adds tests that use already existing tests that write and read collections (all tests pass)



podio: RNTuple backend

TTree based

```
ROOTFrameWriter writer(filename);  
writer.writeFrame(frame);  
writer.finish();
```

```
ROOTFrameReader reader{};  
reader.openFile(filename)  
auto event = podio::Frame(reader.readEntry("events", 0));
```

RNTuple based

```
ROOTNTupleWriter writer(filename);  
writer.writeFrame(frame);  
writer.finish();
```

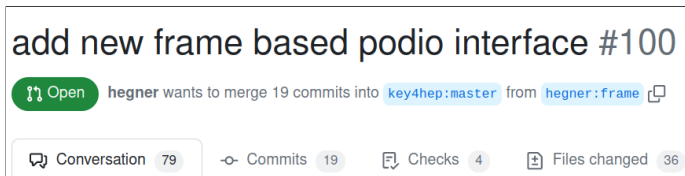
```
ROOTNTupleReader reader{};  
reader.openFile(filename)  
auto event = podio::Frame(reader.readEntry("events", 0));
```

- For the future:
 - Comparisons between the RNTuple and TTree-based backends: reading and writing speed, file size
 - Python bindings for the RNTuple writer and reader

- **Gaudi** based core framework:
 - **k4FWCore** provides the interface between EDMs and Gaudi
 - **k4MarlinWrapper** allows to call Marlin processors
 - **k4SimDelphes** for integration with Delphes
 - **k4SimGeant4** for integration with Geant4
 - **k4Gen** for integration with generators
 - **k4geo** for detector models, previously lcgeo
 - ...

key4hep Framework: Frame Support

- Coming soon, support for `podio::Frame` in `k4FWCore`, Benedikt




- More work needed for the metadata
- DD4hep saves to frames in 1.24 and after (not yet in the release nor in the nightlies)

LCIO to EDM4hep conversion

- Status: Converter EDM4hep to LCIO in <https://github.com/key4hep/k4EDM4hep2LcioConv>
 - Converter from LCIO to EDM4hep: <https://github.com/key4hep/k4LCIOReader>
 - to be replaced by a newer one
 - Leonhard has been using, testing and fixing these extensively:
-
- Support for EventHeader conversion
 - Fixing some broken associations


Add support for EventHeader conversion #13

 Merged tmadlener merged 5 commits into `key4hep:master` from `2ehvogel:event-header-new` on Apr 19

 Conversation 5  Commits 5  Checks 5  Files changed 2

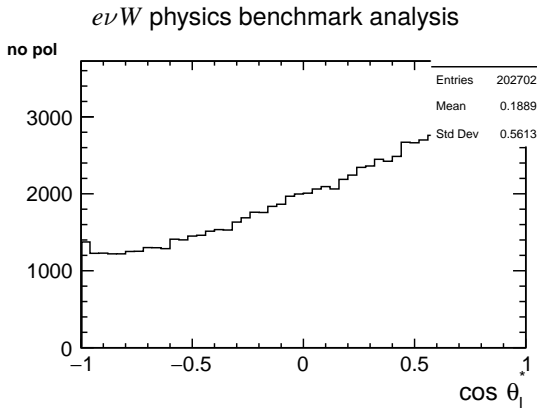
Only put non-subset collections into m_type2cols #32

 Merged andresailer merged 2 commits into `key4hep:master` from `2ehvogel:subset` on May 16

 Conversation 4  Commits 2  Checks 3  Files changed 1

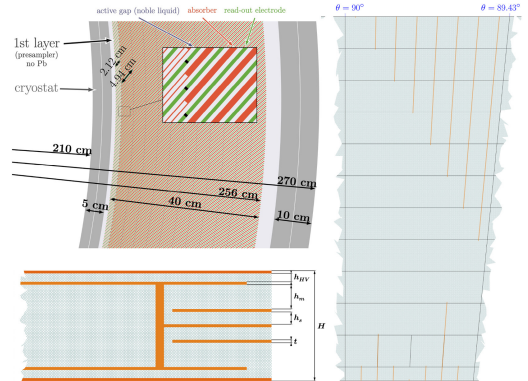
- Leonhard will work on track fitting for electrons with ACTS in Gaudi
- Studying the status of [k4ActsTracking](#), a package to integrate ACTS with key4hep
- User example: using ILD files that are converted to EDM4hep in [jupyter notebooks](#) with functions from FCCAnalyses

```
In [13]: def defines(df):  
         df2 = (df  
               .Define("enu", "get_enu(MCParticles)")  
               .Define("enu_p", "FCCAnalyses::MCParticle::get_p(enu)")  
               .Define("enu_theta", "FCCAnalyses::MCParticle::get_theta(enu)")  
               .Define("e", "enu[0]"))
```



Pandora

- Liquid Argon (LAr) detectors are being studied for future experiments (e.g. FCC)
- Swathi will study jet energy resolution for IDEA-LAr when a full simulation for IDEA is implemented
- Currently studying LAr with CLD (with full simulation)
- While adding the LAr calorimeter inside CLD overlaps were found so changes in the geometry had to be done
- Implementing a Gaudi algorithm to use Pandora PFA with Gaudi



key4hep Builds

- Builds are done in containers for Centos 7, AlmaLinux 9 and Ubuntu 22.04
- Users only need `source /cvmfs/sw-nightlies.hsf.org/key4hep/releases`

```
{jcarcell@lxplus924.cern.ch} latest % ls -l /cvmfs/sw-nightlies.hsf.org/key4hep/releases/latest
total 2
lrwxrwxrwx. 1 cvmfs cvmfs 87 Jun 20 02:15 x86_64-almalinux9-gcc11.3.1-opt → /cvmfs/sw-nightlies.hsf.org/key4hep/releases/2023-06-20/x86_64-almalinux9-gcc11.3.1-opt
lrwxrwxrwx. 1 cvmfs cvmfs 84 Jun 20 02:13 x86_64-centos7-gcc12.2.0-opt → /cvmfs/sw-nightlies.hsf.org/key4hep/releases/2023-06-20/x86_64-centos7-gcc12.2.0-opt
lrwxrwxrwx. 1 cvmfs cvmfs 88 Jun 11 21:46 x86_64-ubuntu22.04-gcc11.3.0-opt → /cvmfs/sw-nightlies.hsf.org/key4hep/releases/2023-06-10/x86_64-ubuntu22.04-gcc11.3.0-opt
```


- Next release will have support for Centos 7, AlmaLinux9 and Ubuntu 22.04
- Constant communication with users: many of the issues are solved for the next nightly (in less than 24 hours)

key4hep Builds: Spack

- Status: spack is used to make the builds and then they are copied to cvmfs
- We used to have a fork of spack
 - We were 3000 or 4000 commits behind spack develop
 - Changes in upstream but not in our fork
 - Changes in our fork but not in upstream
 - Rebasing was very painful
- Fork archived
- Keeping up with Spack by updating regularly
- Updating was a bit painful, now libraries are opt-in to be added to LD_LIBRARY_PATH



forked from [spack/spack](#)

 key4hep ▾

 12 branches

 33 tags

This branch is [41 commits ahead](#), [1175 commits behind](#) spack:develop.

key4hep Builds: Nightlies

- Nightlies were updated
- Two build types depending if the build is done from scratch or not
- Builds from scratch will get the latest packages and let us know that the changes in spack don't break our builds - Updates every O(weeks)
- Daily builds that use as upstream the builds from scratch, they only build the packages that have changed (or those that depend on a package that has changed)
- Three new hidden files in each release
 - `.scratch`: If it's there, that means it's a build from scratch (all packages)
 - `.spack-commit`: Which commit of spack was used to build this release
 - `.key4hep-spack-commit`: Which commit of key4hep-spack was used to build this release

key4hep Builds: Tests and Next Steps

- New **usability tests** are being added: compilation, ROOT, python, python packages, whizard, key4hep tools
- These tests come from experience, mainly from what people use that one day doesn't work, to make sure it doesn't repeat

```
cat > ee.sin <<EOF
process ee = e1, E1 => e2, E2
sqrts = 360 GeV
n_events = 10
sample_format = lhef
simulate (ee)
EOF
run_test "whizard test" "whizard -r ee.sin"
```

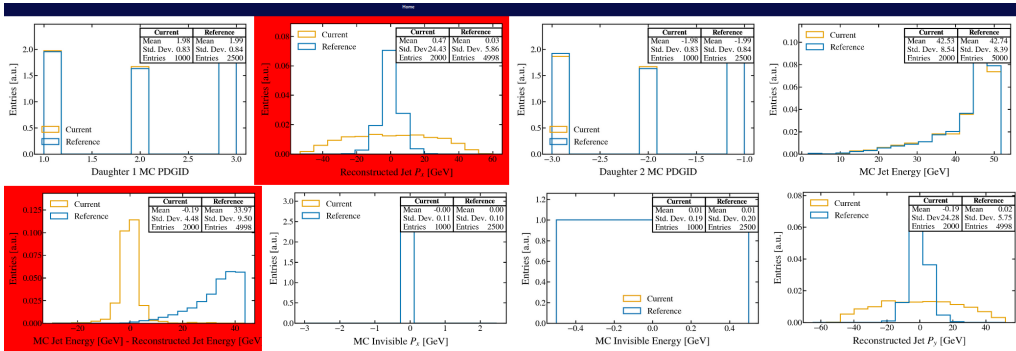
- Next steps:
 - Use build caches to deploy to cvmfs
 - GCC 13 and C++20
 - clang build
 - MacOS build

key4hep Validation: Simulation and Reconstruction

- Check the simulation and reconstruction chain
- Run daily, use the latest key4hep nightlies
- Run a simulation with DD4hep, then reconstruction, then analysis scripts and then make plots
- Results are compared to a reference sample
- Plots are deployed to WebEOS (static webpage)
- <https://key4hep-validation.web.cern.ch/>
- Work in progress, no documentation yet

key4hep Validation: The Webpage



- Checks (are) will be done to check if the distributions are too different and then make the plot background red, for example
- This is Z to qq at 100 GeV




Validation Example: DD4hep

- This PR fixes hits being dropped

LCIOConversions: fix attaching SimCaloHit collections to event :

 Merged andresailer merged 1 commit into [AIDASoft:master](#) from [andresailer:fixLcioCalos](#)  on Nov 17, 2022


Conversation 1 Commits 1 Checks 13 Files changed 1

 andresailer commented on Nov 16, 2022 Member ...

BEGINRELEASENOTES

- LCIOConversions: fix attaching SimCaloHit collections to event, this was accidentally dropped in [Allow re-using readouts for different detectors](#) #922

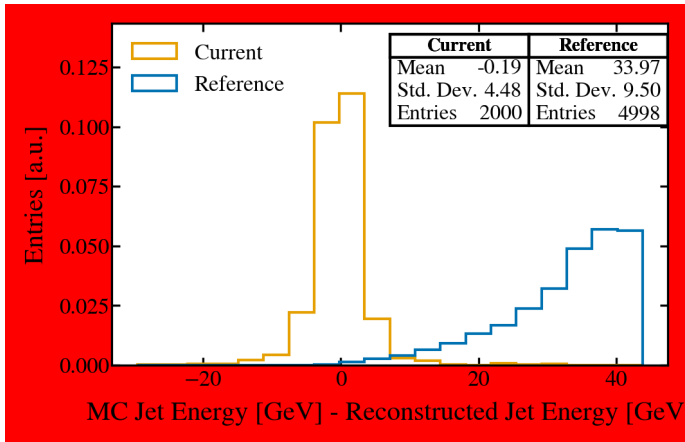
ENDRELEASENOTES



- This happened after 1.23
- DD4hep has being pinned to 1.23 for a while in the nightlies not to save output files in the Frame format
- We didn't have the fix until it was noticed that something was looking wrong

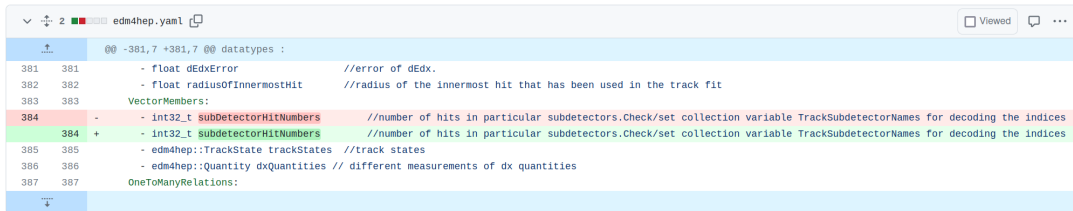
Validation Example: DD4hep

- If now we use the validation where a reference sample has the bug (note that in the future the reference sample will be a "good" sample) and the current has it fixed it's very obvious:



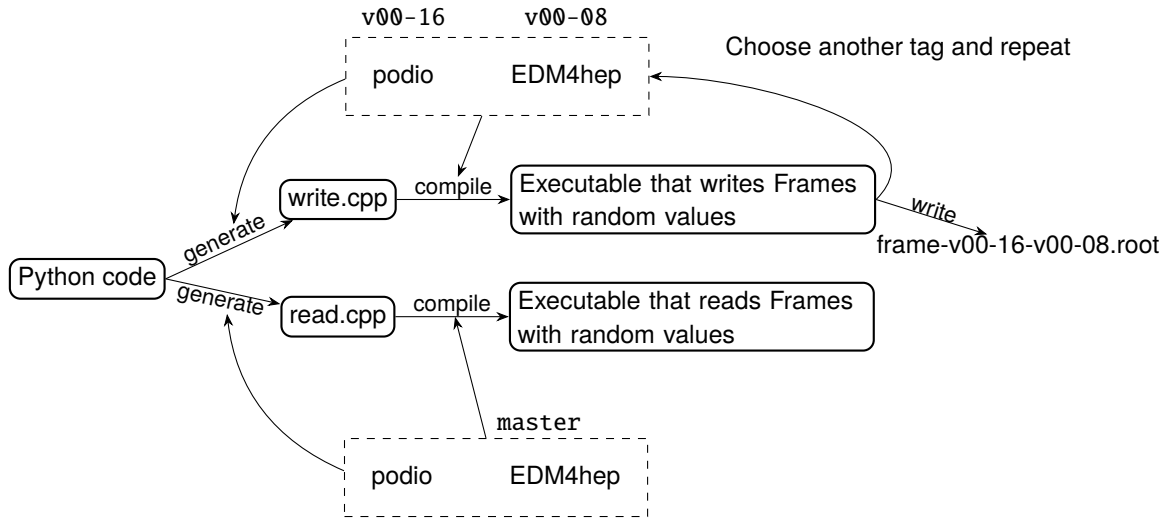
key4hep Validation: EDM4hep

- Recently a new kind of validation was added
- Check what happens if we try to read old files produced with an old version of podio and EDM4hep
- For example, recently we had a change, a simple rename, can we read fine files created before the rename?



```
edm4hep.yaml
@@ -381,7 +381,7 @@ datatypes :
381 381     - float dEdxError           //error of dEdx.
382 382     - float radiusOfInnermostHit //radius of the innermost hit that has been used in the track fit
383 383     VectorMembers:
384 384     - - int32_t subdetectorHitNumbers //number of hits in particular subdetectors.Check/set collection variable TrackSubdetectorNames for decoding the indices
385 385     + - int32_t subdetectorHitNumbers //number of hits in particular subdetectors.Check/set collection variable TrackSubdetectorNames for decoding the indices
386 386     - edm4hep::TrackState trackStates //track states
387 387     - edm4hep::Quantity dxQuantities // different measurements of dx quantities
387 387     OneToManyRelations:
```

key4hep Validation: EDM4hep



key4hep Validation: EDM4hep

write.cpp

```
auto zimhm = edm4hep::EventHeaderCollection();
auto hlmb1 = zimhm.create();
hlmb1.setEventNumber(50);
hlmb1.setRunNumber(42);
hlmb1.setTimeStamp(77);
hlmb1.setWeight(37);
frame.put(std::move(zimhm), "EventHeaderCollection");
```

read.cpp

```
auto& zimhm = frame.get<edm4hep::EventHeaderCollection>("EventHeaderCollection");
auto hlmb1 = zimhm[0];
if (hlmb1.getEventNumber() != int32_t(50)) {
    std::cout << "Error: hlmb1.getEventNumber() != 50" << std::endl;
    ret = 1;
}
...
```

- Generated frames, first the podio version, then the EDM4hep version

```
$ ls -lh
86K Jun 19 15:03 frame-v00-16-01-v00-07.root
86K Jun 19 15:04 frame-v00-16-02-v00-07.root
94K Jun 19 15:06 frame-v00-16-03-v00-07.root
94K Jun 19 15:08 frame-v00-16-04-v00-07.root
94K Jun 19 14:54 frame-v00-16-05-v00-07-01.root
94K Jun 19 14:54 frame-v00-16-05-v00-07-02.root
94K Jun 19 15:09 frame-v00-16-05-v00-07.root
112K Jun 19 14:56 frame-v00-16-05-v00-08.root
112K Jun 19 14:57 frame-v00-16-05-v00-09.root
86K Jun 19 15:02 frame-v00-16-v00-07.root
```

- Results under study

Summary

- Progress in Key4hep in different areas. Addition of new people has helped
- The community of Key4hep users, developers keeps getting bigger → improvements → more users, developers
- Increasing the number of features while at the same time making it more robust
- **Lots of new developments soon!**

Backup

The Validation Webpage

Home

This is a webpage for validation of Key4hep software. The validation is done automatically and runs in Gitlab CI. The results are stored in EOS and published to this webpage. For selecting different detectors and geometries, click the button below to expand a list of available options.

Simulation and reconstruction

Detector, geometry and process

[CLIC_o3_v15](#)

Last updated: 2023-06-13 10:09:58

The Validation Webpage

Home	Jet Validation
key4hep-spack	a19c7cc6fd488a5db3e446d842a0bdd82c3e15f6
spack	None
nightly	/cvmfs/sw-nightlies.hsf.org/key4hep/releases/2023-06-10/x86_64-almalinux9-gcc11.3.1-opt

Datamodel Validation: Generated Code

write.cpp

```
auto zimhm = edm4hep::EventHeaderCollection();
auto hlmb1 = zimhm.create();
hlmb1.setEventNumber(50);
hlmb1.setRunNumber(42);
hlmb1.setTimeStamp(77);
hlmb1.setWeight(37);
frame.put(std::move(zimhm), "EventHeaderCollection");
```

read.cpp

```
auto& zimhm = frame.get<edm4hep::EventHeaderCollection>("EventHeaderCollection");
auto hlmb1 = zimhm[0];
if (hlmb1.getEventNumber() != int32_t(50)) {
    std::cout << "Error: hlmb1.getEventNumber() != 50" << std::endl;
    ret = 1;
}
if (hlmb1.getRunNumber() != int32_t(42)) {
    std::cout << "Error: hlmb1.getRunNumber() != 42" << std::endl;
    ret = 1;
}
if (hlmb1.getTimeStamp() != uint64_t(77)) {
    std::cout << "Error: hlmb1.getTimeStamp() != 77" << std::endl;
    ret = 1;
}
if (hlmb1.getWeight() != float(37)) {
    std::cout << "Error: hlmb1.getWeight() != 37" << std::endl;
    ret = 1;
}
```

Datamodel Validation: Generated Code

- Also supports vector members

write.cpp

```
auto ycjbb = edm4hep::TrackCollection();
auto gghbj = ycjbb.create();
...
gghbj.addToDxQuantities({ 38, 47, 23 });
```

read.cpp

```
if (gghbj.getDxQuantities(0).type != edm4hep::Quantity({ 38, 47, 23 }).type) {
    std::cout << "Error: gghbj.getDxQuantities(0) != { 38, 47, 23 }" << std::endl;
    ret = 1;
}
if (gghbj.getDxQuantities(0).value != edm4hep::Quantity({ 38, 47, 23 }).value) {
    std::cout << "Error: gghbj.getDxQuantities(0) != { 38, 47, 23 }" << std::endl;
    ret = 1;
}
if (gghbj.getDxQuantities(0).error != edm4hep::Quantity({ 38, 47, 23 }).error) {
    std::cout << "Error: gghbj.getDxQuantities(0) != { 38, 47, 23 }" << std::endl;
    ret = 1;
}
```

Datamodel Validation: Next Steps

- Python bindings? Not for writing since old versions of EDM4hep don't have it, possibly for reading since we use the nightlies
- Other datamodels? In principle possible and possibly few changes are needed but the python script needs rewriting some logic in a better way
- Add to the key4hep-validation webpage?
- Support relations
- Reuse podio parser?