# Rivet monthly dev meeting

2 December 2020

## Headlines

- **Rivet 3.1.3 + YODA 1.8.5 released**
  - YODA 1.8.4 earlier in Nov
  - Lots of fixes, and UI improvements for multiweights
  - Most Dockers built and deployed; rivet-herwig held up by Herwig bootstrap error
  - Argh: build stability issues, due to analysis concat behavioural variations **3.1.4??**
- **Re-activating enthusiasm / WG activities**
  - HD consistency, plotting, statistics
  - 16-17 Dec: Christmas hackathon in Scotland
  - Outreach to ALICE and UK HEP (CB, JMB, AB) in Nov)
  - Christian B organised a useful discussion with EIC re. collaboration areas
- **Standardising: weight names and event-record content**
  - Proposal drafted by Chris G and AB
  - Plan to circulate to MCnet management & LHC expts
  - Follow-up meeting… in Jan?

BACKUP

# Release plan

- ~~**Rivet 3.1.3**~~
  - ~~Fixes and improvements: new analyses, util functions, Doxy cleaning, … review~~
    - ~~weight-subset improvements & bugfix, nominal-weight specification~~
    - ~~logic fixes & C++ improvements in “higher order” select/discard + sortBy~~
    - ~~take DressedLepton origin vertex position from bare lepton~~
    - ~~MC-only kT splittings broken in Rivet3 but fixed on release branch~~
    - ~~mkhtml JS filtering!~~
    - ~~CB/JMB/LL: DISKinematics issue??~~
    - ~~AB: add jet filtering feature; avoid MET=0 peak in low-MET$_{\text{true}}$ smearing~~
    - ~~CG: rivet-merge broken? Add _SQRTS to output? yodamerge scaling/speed~~

- **Longer-term, toward 3.2.0**
    - finish Aditya & Nick performance and YODA API work, add HDF5 ana-data
    - beam-check consistency and enum rationalising
    - FastJets(FinalState) -> FastJets(ParticleBaseFinder)

- **Let’s avoid a pre-Xmas release rush, for once! Eeek**

## Major work plans

- **Convert finalize output to “dead” objects**
  - Final objects really will mean “what was plotted/listed in the paper”
  - Allow eager conversion to solve “no-bin-width issue”
  - Best that we wait for binned measurement YODA2 types: no more scatters!

- **HDF5 analysis data machinery (Holger)** *Status?*
  - Also interested in HepMC and YODA HDF5 formats
  - Holger to ping CMS, prototype interface

- **Plotting (Christian B et al)**
  - Plan: generate Python MPL scripts *without* TeX, .plot styles → YAML
  - Rivet labels tested: MathText fails due to missing std symbols. Can we extend?
  - Stalled for a while… restarting? Possible student help from David Grellscheid
  - Christian to prototype the Python-script generation
  - Chris to extract weight-handling logic from rivet-cmphistos

## Performance in Rivet and YODA (Aditya Kumar, AB)

- **Profiling revealed bottlenecks: thanks Aditya!**
  - HepMC ASCII I/O (obviously) — taken out of tests by event-reuse
  - GenEvent copying — for sanitising, but hardly used: removed from Rivet.
    Could/should generators write smaller "essential" events by default?
    Awkwardness: we still normalise GenEvent units… so not quite analysing a const GenEvent.
    But can't justify an expensive copy for *unit conversion*…
  - PID functions — sped up charge lookups by special-cases. Marginal gain
  - Multiweight calls to histo fill() *very* expensive: ~40-50% CPU!
    100+ consecutive fills with same *x*: tried caching in YODA but no benefit:
    cache-check costs the same as linear bin lookup! *Maybe cache in Rivet?*

- **Thread-safety.** *"Just store a ProjectionHandler in AnalysisHandler: easy!"...?*
  - But then who do Projection constructors (recursively) register their contained projections with, before they themselves have been bound to a PH?
  - "Declare queue" implemented: not yet working (thx, unique_ptr), but should do
    *What* should *the Projection ownership be?!*

# YODA generalised datatypes (Nick Rozinsky, LC, AB)

- **Long-understood limitations of YODA types and design**
  - Overreach in attempted non-factorisable binnings: composed 1D axes are fine
  - Complexity/mess in 2D overflows: need “infinity binning”
  - Need for binned “dead” data objects… or any type, actually
  - Want programmatic access to axis number and global/local bin indexing
  - Want labelled/discrete binnings as well as continuous
  - Code duplication, particularly in Cython interface building

- **Major YODA redesign using modern C++ magic. Thanks Nick!**
  - E.g. Histo1D → wrapper of a BinnedStorage<CAxis, Dbn<1>> + sugar
  - + arbitrary mixtures, e.g. 3D binnings of doubles, discretely labelled counters, …
  - Adaptors used to map fill/set behaviours. Can do the same for I/O read/write?

- **Path to a YODA2 release:**
  - Needs I/O adaptors and user-facing refinements. Tie in with HDF5 format?