The ATLAS Event Data Model

Attila Krasznahorkay

Event Data Model?

- Is the collection of classes and associated functions that we use to describe an “event” in the ATLAS detector
    - The “event” in question may be the output of the event generator, the G4 simulation or the raw data coming from the detector
    - But in this talk we will only focus on the “analysis event data model”, the way in which we describe the output of the reconstruction
- I.e. we will discuss how we store “electrons”, “muons”, “jets”, etc., and how you can use them during your analysis

Why an Event Data Model?

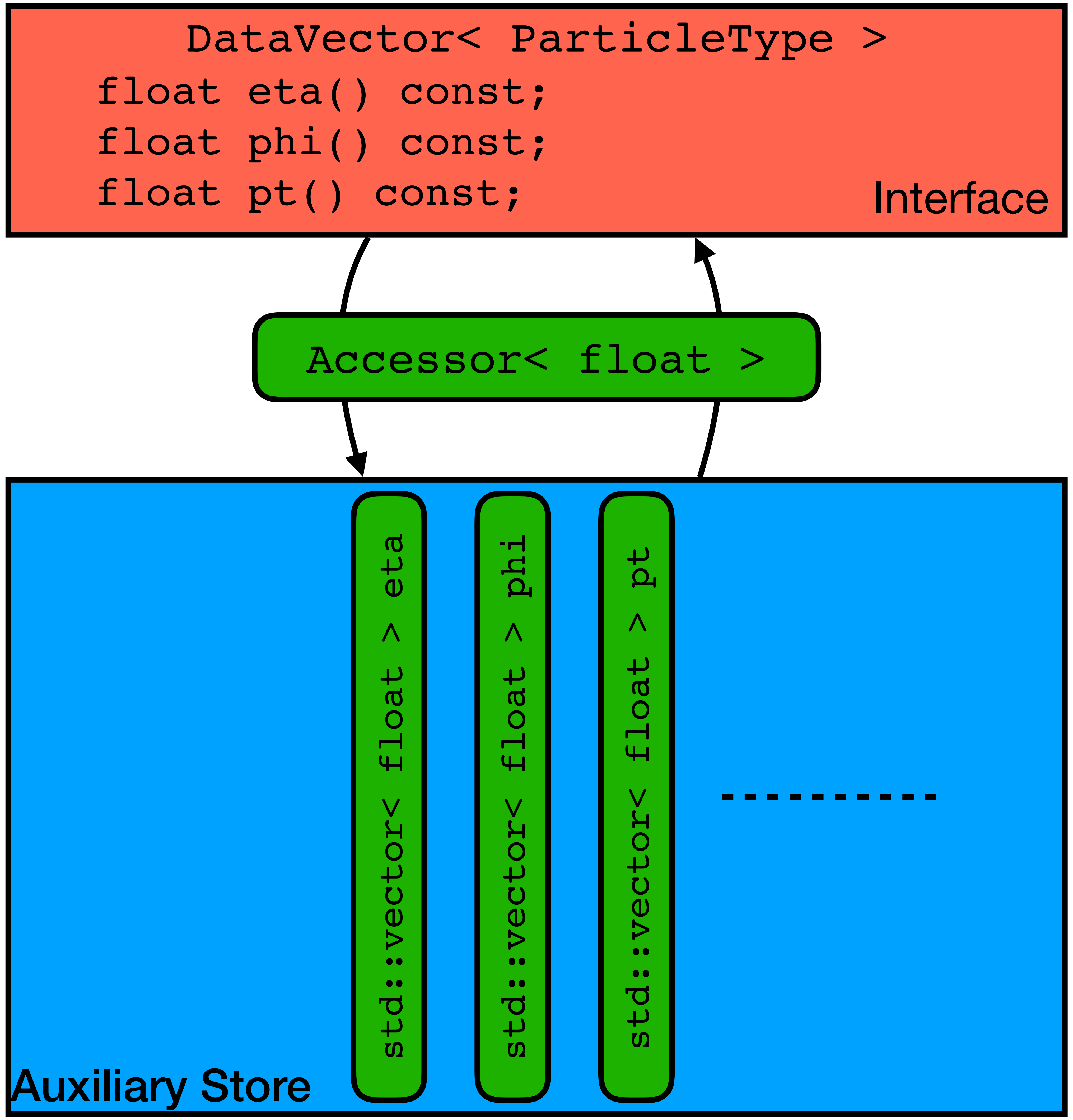
- It is in a large part a software design question…
- By defining a single way in which objects are described in our code, we ensure that
  - Independent components can conveniently exchange information with each other
    - For instance that the output of the inner detector tracking can be used as-is by the electron reconstruction
  - Analysis tools can be developed around a single description of the events, making it easier to start using a tool from scratch
  - We only have to implement things efficiently / correctly once
    - Some of the code behind the scenes is not completely trivial. It would be a shame to implement it multiple times.

The xAOD EDM

- Is the EDM that we developed for LHC’s Run 2
  - The EDM used for the reconstruction output during Run 1 was not appropriate for direct user analysis, and hence practically nobody used it directly
- During Long Shutdown 1 we set out to design an EDM fulfilling all the following requirements:
  - Be possible to read and write with both Athena, and much lighter-weight software independent of Athena
  - Provide convenience features important for both reconstruction and analysis, including adding “named variables” to objects as easily as possible
  - Be able to provide all the features (“slimming”, “thinning”, etc.) needed by the derivation framework
    - To be discussed later on in the week…
  - Provide the most efficient code for manipulating event data that we could

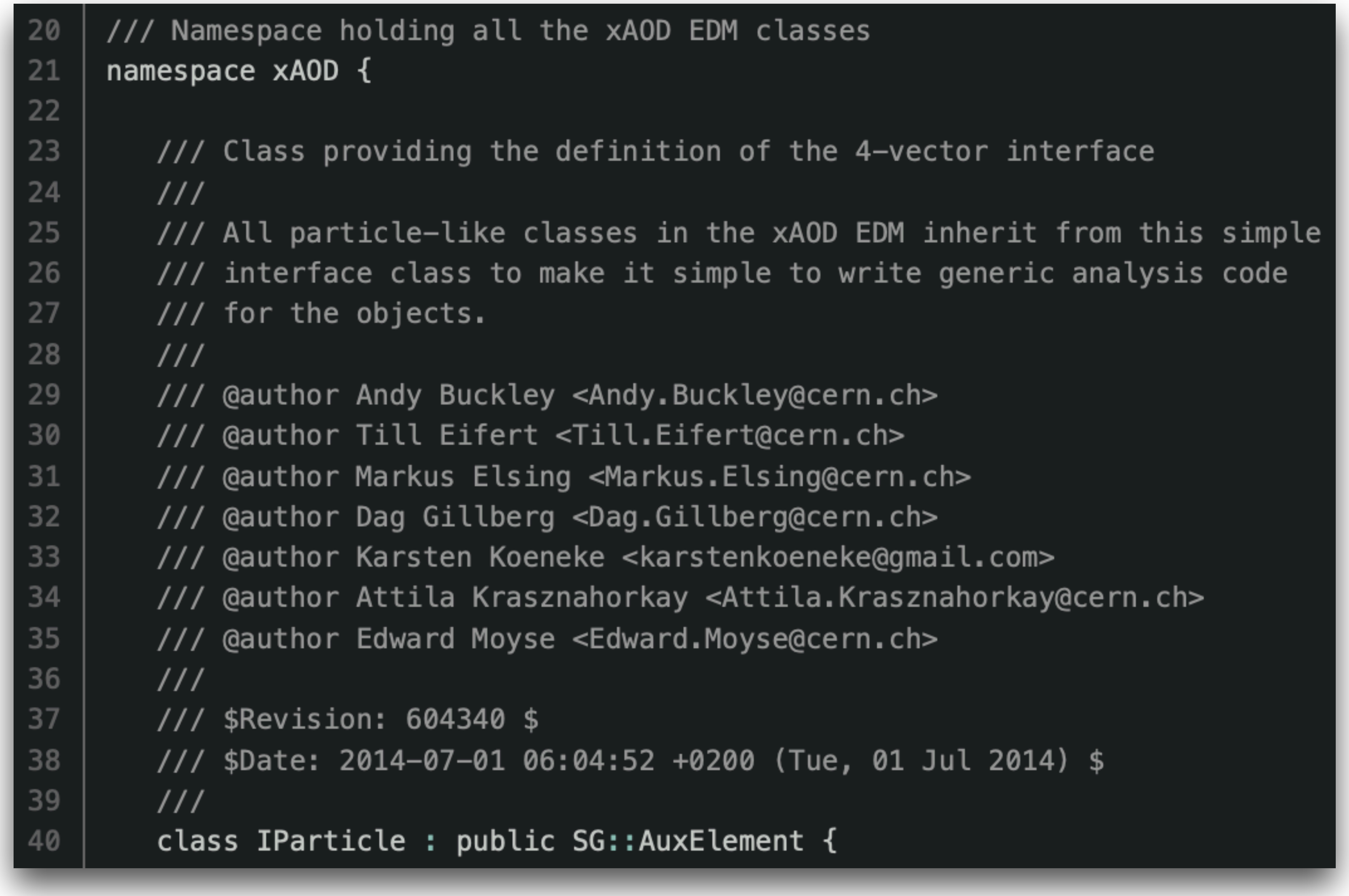
The xAOD EDM

- The code separates the interfaces that you interact with, from the objects (auxiliary stores) that physically hold on to the data
  - You need to be aware of this design, but don't need to know the implementation in much more detail than this
- This abstraction between the user interface and the storage implementation allows us to do a **lot** of clever tricks with our data 😉

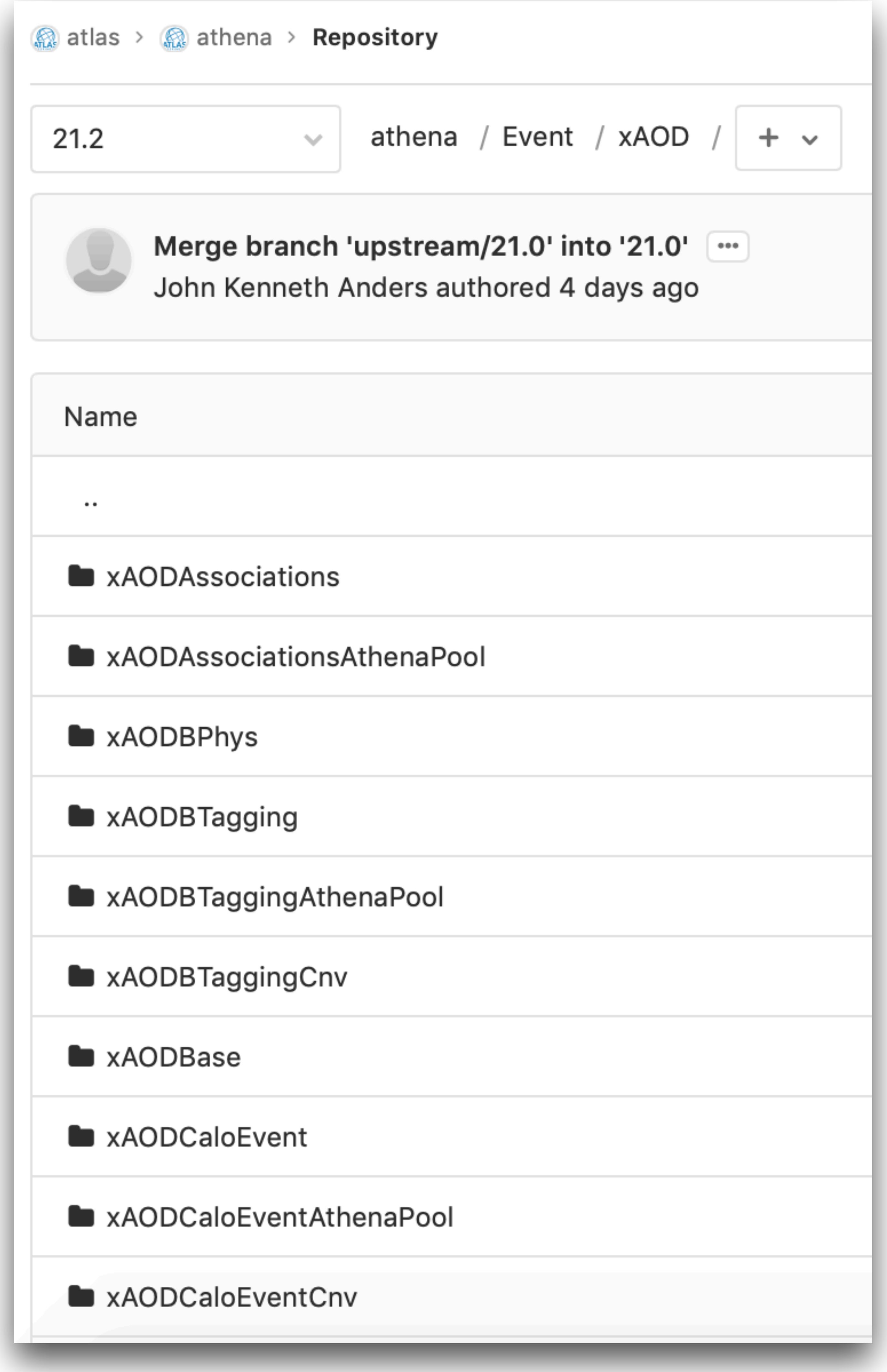


The xAOD EDM

- All of the code lives under Event/xAOD in the offline software repository
- All of the classes — that you normally need to interact with — are defined in the xAOD namespace



```
/// Namespace holding all the xAOD EDM classes
namespace xAOD {

   /// Class providing the definition of the 4-vector interface
   ///
   /// All particle-like classes in the xAOD EDM inherit from this simple
   /// interface class to make it simple to write generic analysis code
   /// for the objects.
   ///
   /// @author Andy Buckley <Andy.Buckley@cern.ch>
   /// @author Till Eifert <Till.Eifert@cern.ch>
   /// @author Markus Elsing <Markus.Elsing@cern.ch>
   /// @author Dag Gillberg <Dag.Gillberg@cern.ch>
   /// @author Karsten Koeneke <karstenkoeneke@gmail.com>
   /// @author Attila Krasznahorkay <Attila.Krasznahorkay@cern.ch>
   /// @author Edward Moyse <Edward.Moyse@cern.ch>
   ///
   /// $Revision: 604340 $
   /// $Date: 2014-07-01 06:04:52 +0200 (Tue, 01 Jul 2014) $
   ///
   class IParticle : public SG::AuxElement {
```



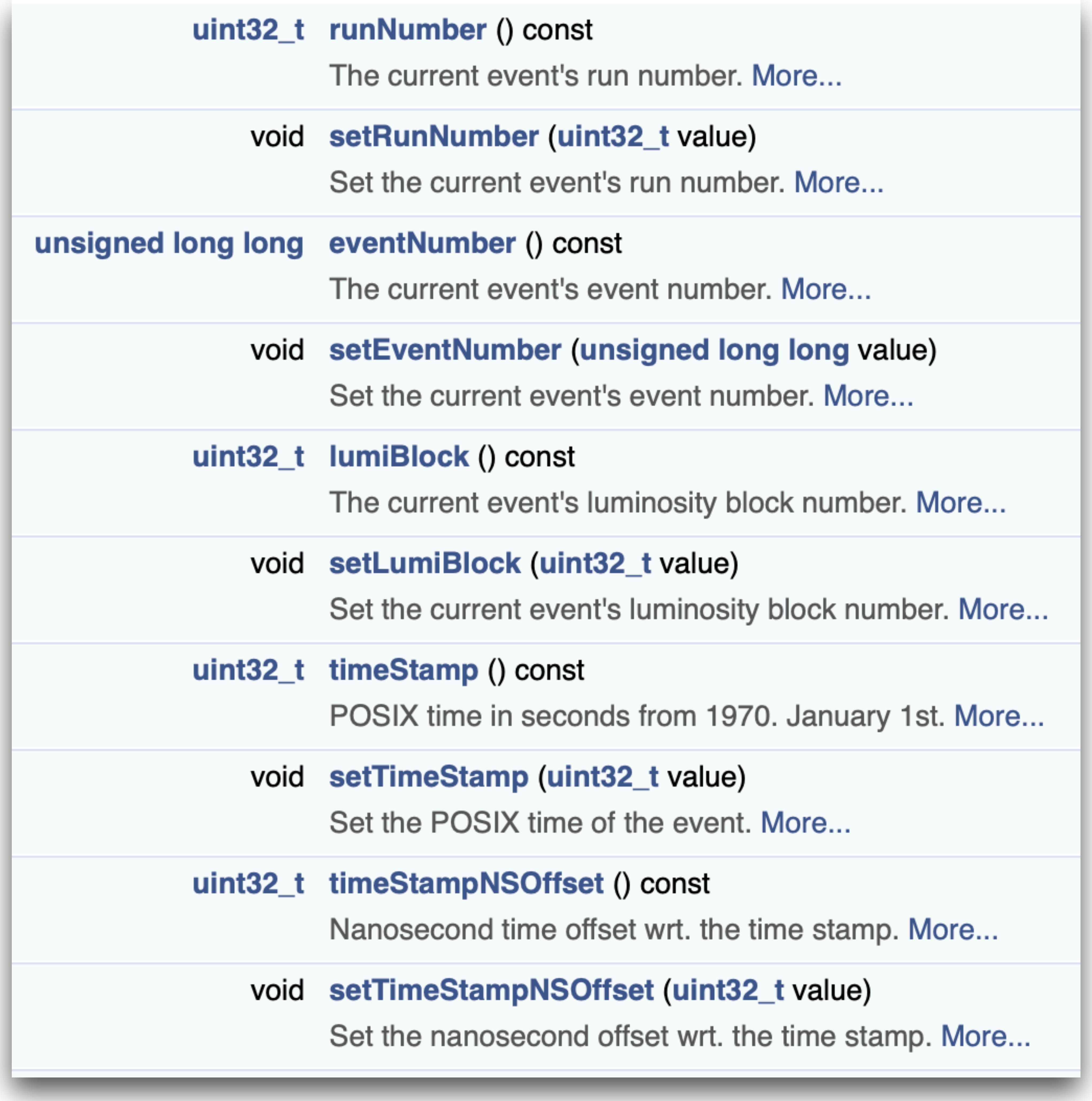
atlas › athena › **Repository**

21.2 | athena / Event / xAOD / | +

Merge branch 'upstream/21.0' into '21.0' ···
John Kenneth Anders authored 4 days ago

Name
..
xAODAssociations
xAODAssociationsAthenaPool
xAODBPhys
xAODBTagging
xAODBTaggingAthenaPool
xAODBTaggingCnv
xAODBase
xAODCaloEvent
xAODCaloEventAthenaPool
xAODCaloEventCnv

xAOD::EventInfo

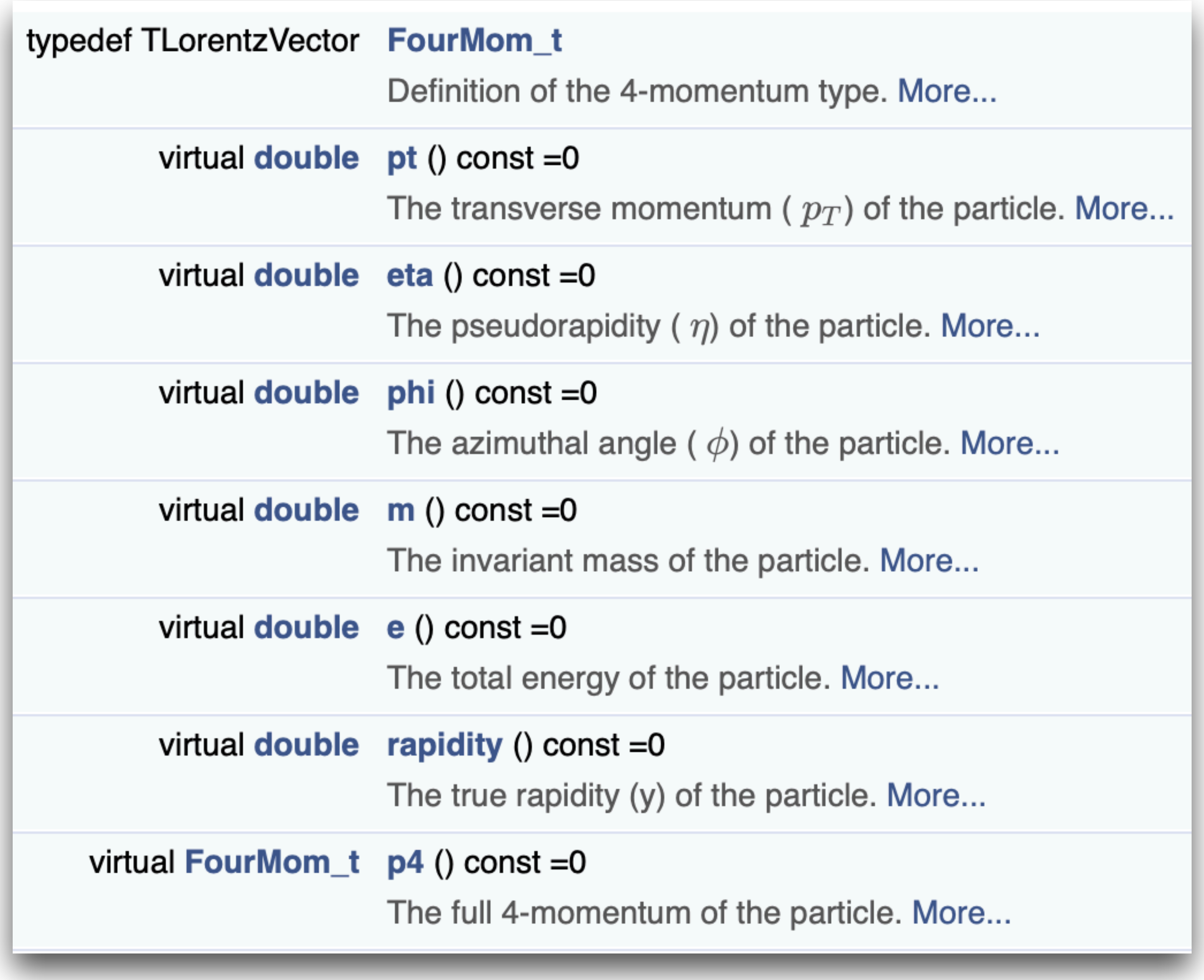


uint32_t	runNumber () const The current event's run number. More...
void	setRunNumber (uint32_t value) Set the current event's run number. More...
unsigned long long	eventNumber () const The current event's event number. More...
void	setEventNumber (unsigned long long value) Set the current event's event number. More...
uint32_t	lumiBlock () const The current event's luminosity block number. More...
void	setLumiBlock (uint32_t value) Set the current event's luminosity block number. More...
uint32_t	timeStamp () const POSIX time in seconds from 1970. January 1st. More...
void	setTimeStamp (uint32_t value) Set the POSIX time of the event. More...
uint32_t	timeStampNSOffset () const Nanosecond time offset wrt. the time stamp. More...
void	setTimeStampNSOffset (uint32_t value) Set the nanosecond offset wrt. the time stamp. More...

- Holds overall information about the current event
  - The event identifiers (run number, event number, luminosity block number)
  - The time at which the event was taken
  - Data taking conditions (pileup…)
  - Detector status flags
  - Trigger/DAQ stream information
  - etc.
- Is usually one of the first objects that you will want to access in your analysis code

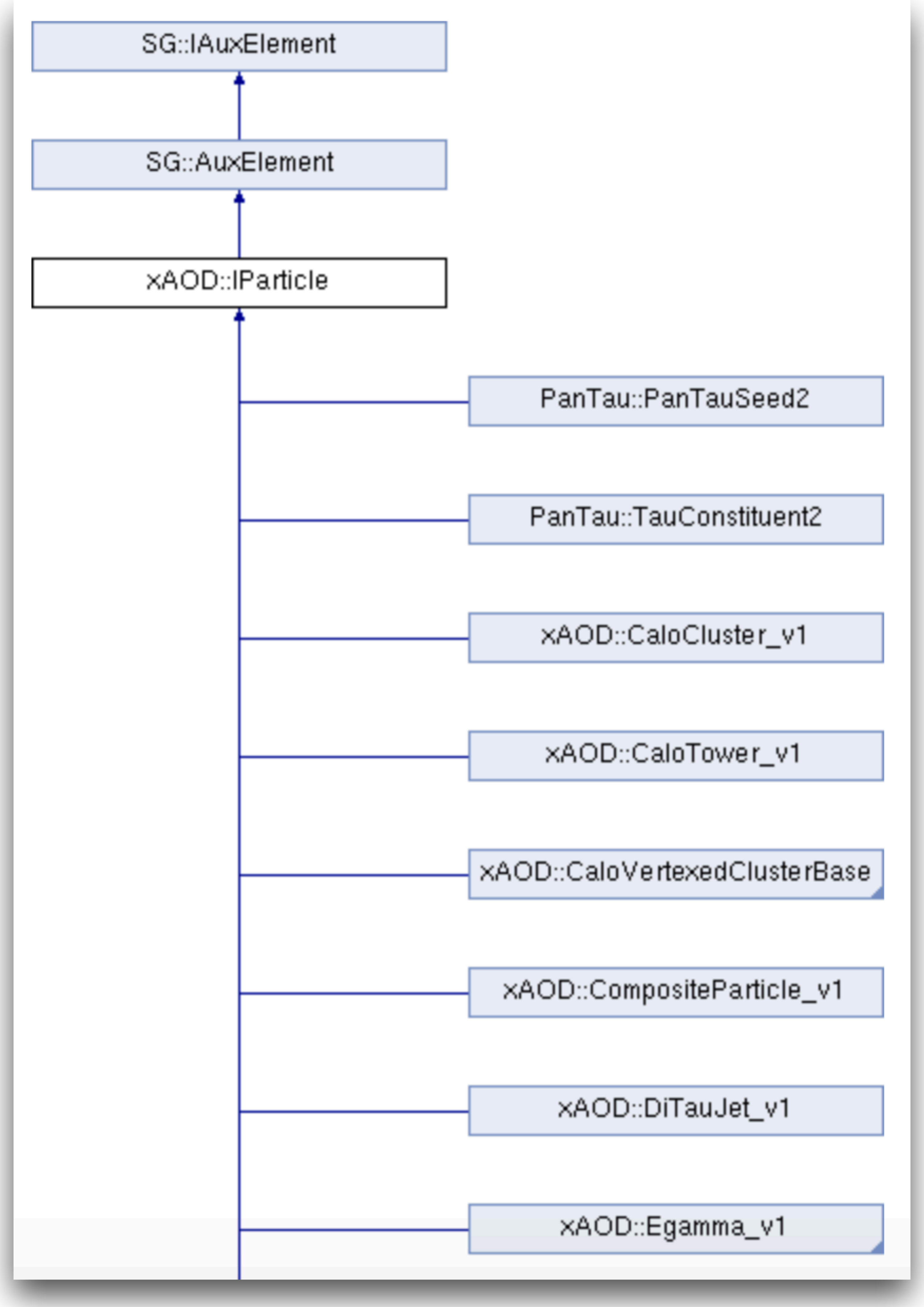
xAOD::IParticle

- All “particle-like” classes implement the same interface
  - Makes sure that all such objects would behave consistently
  - Allows analysis code to treat certain objects uniformly, nor caring about their exact type
- Most of the xAOD types implement this interface
  - The main exceptions are `xAOD::MissingET` and a number of trigger classes



typedef TLorentzVector	**FourMom_t** Definition of the 4-momentum type. More...
virtual **double**	**pt** () const =0 The transverse momentum (p_T) of the particle. More...
virtual **double**	**eta** () const =0 The pseudorapidity (η) of the particle. More...
virtual **double**	**phi** () const =0 The azimuthal angle (ϕ) of the particle. More...
virtual **double**	**m** () const =0 The invariant mass of the particle. More...
virtual **double**	**e** () const =0 The total energy of the particle. More...
virtual **double**	**rapidity** () const =0 The true rapidity (y) of the particle. More...
virtual **FourMom_t**	**p4** () const =0 The full 4-momentum of the particle. More...

Versioning



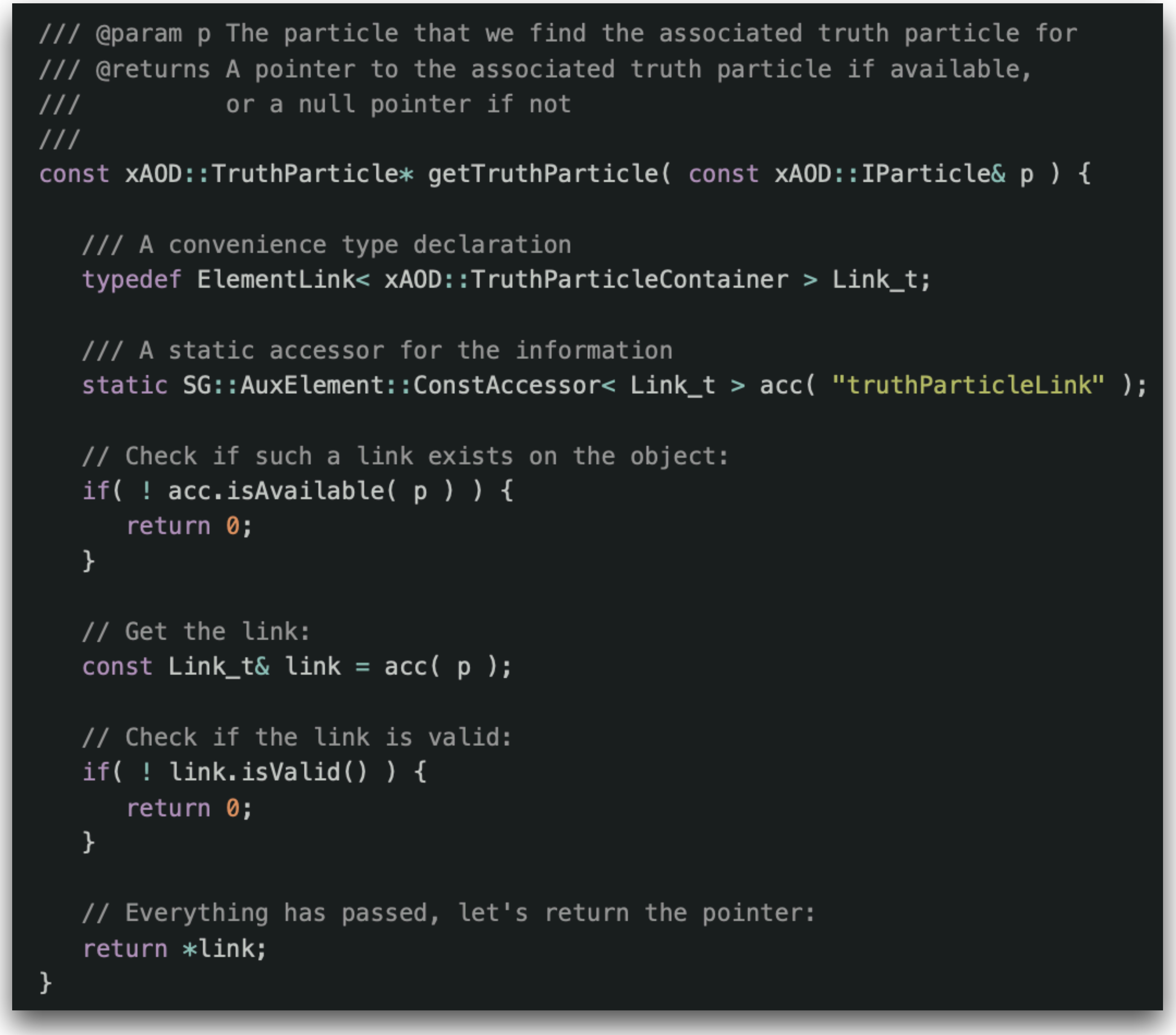
- Things are a bit more complicated… ☹
- Both the interface classes and the auxiliary store classes are “versioned”
  - This is something that we need for “schema evolution”
    - That is when we change how a given object is described in the software. This versioning allows the code to still read files written with the “old” format, and present the objects in the “new” format to the code
  - Again, this is something that you need to know about, but do not need to know all the details of
- Your code should **never** use these versioned names, but only the non-versioned ones
  - Names like “`xAOD::Electron`” are typedef-s of the latest version of the class

Containers

- Most objects are stored in containers
  - Most objects we have a variable number of
  - But of course there are exceptions, xAOD::EventInfo is the most obvious
- We have dedicated type for this, called `DataVector`
  - This custom vector type is the one implementing the interface `<->` auxiliary store separation
- `DataVector<T>` has an interface mimicking `std::vector<T*>`
  - If we were to design the code from scratch today, we would probably do it differently. But the original design was meant for C++ compilers available around 2000…
- After the auxiliary store implementation the second biggest feature is that containers mimick the inheritance of the objects that they wrap
  - So for instance `DataVector<xAOD::Electron>` can be used as `DataVector<xAOD::IParticle>` in the code!
- We provide convenient type names for the containers, like `xAOD::MuonContainer`, which are just typedef-s to — in this case — `DataVector<xAOD::Muon>`

Object Links

- Most of the objects are not completely standalone
  - Muons point to the track particles that they were reconstructed from, electrons point to the track(s) and the calorimeter cluster(s) that they were reconstructed from, etc.
- These relationships are implemented by a custom type, ElementLink
  - We can not just use “bare pointers” to save these relationships, as ROOT would not be able to handle such an EDM efficiently
- `ElementLink<DataVector<T> >` behaves as a smart pointer to `T*`
- In most cases you will not need to interact directly with such objects. We provide helper functions in our code to provide you with simple object pointers.

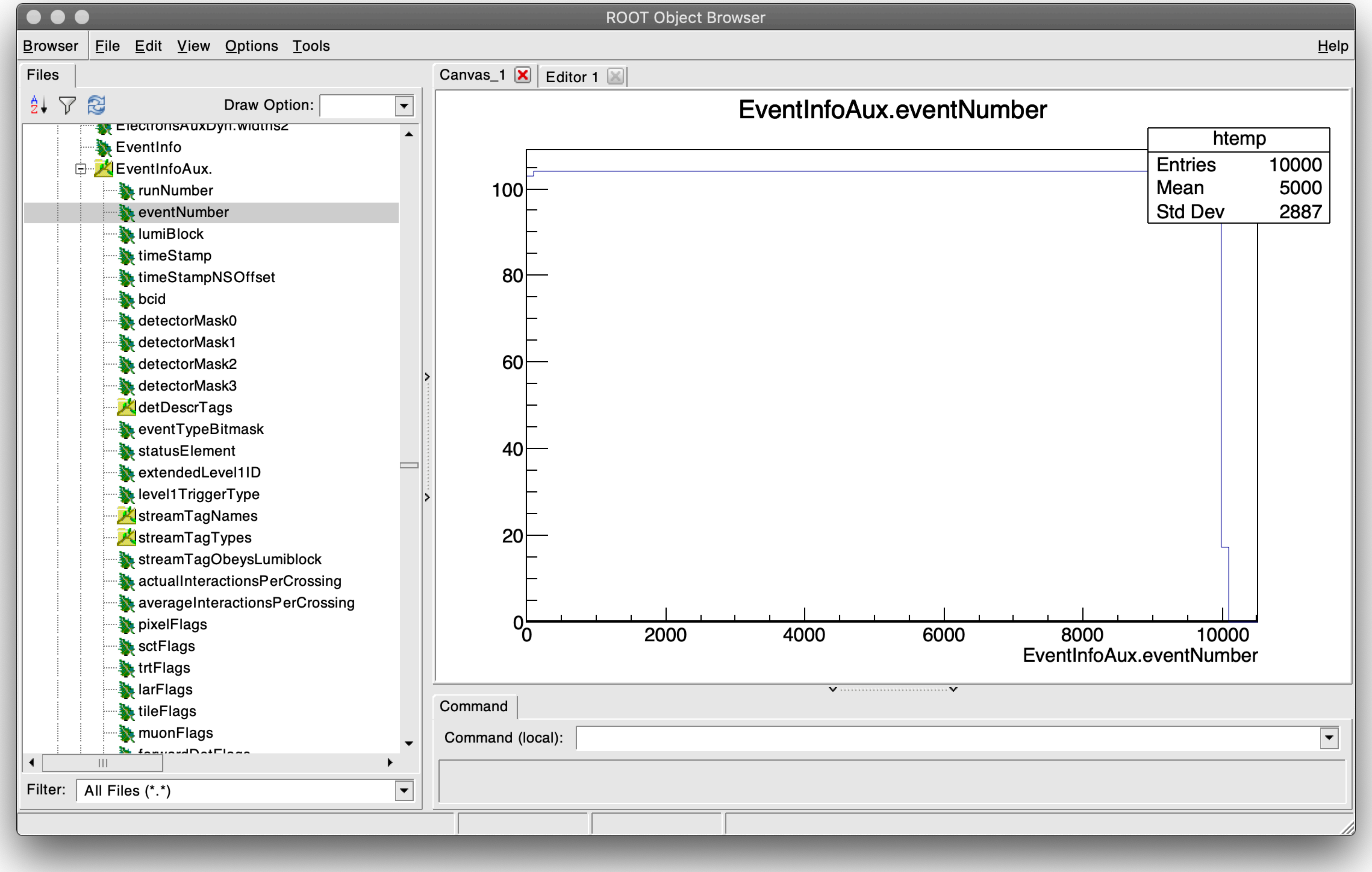


```
/// @param p The particle that we find the associated truth particle for
/// @returns A pointer to the associated truth particle if available,
///          or a null pointer if not
///
const xAOD::TruthParticle* getTruthParticle( const xAOD::IParticle& p ) {

   /// A convenience type declaration
   typedef ElementLink< xAOD::TruthParticleContainer > Link_t;

   /// A static accessor for the information
   static SG::AuxElement::ConstAccessor< Link_t > acc( "truthParticleLink" );

   // Check if such a link exists on the object:
   if( ! acc.isAvailable( p ) ) {
      return 0;
   }

   // Get the link:
   const Link_t& link = acc( p );

   // Check if the link is valid:
   if( ! link.isValid() ) {
      return 0;
   }

   // Everything has passed, let's return the pointer:
   return *link;
}
```

Shallow Copies

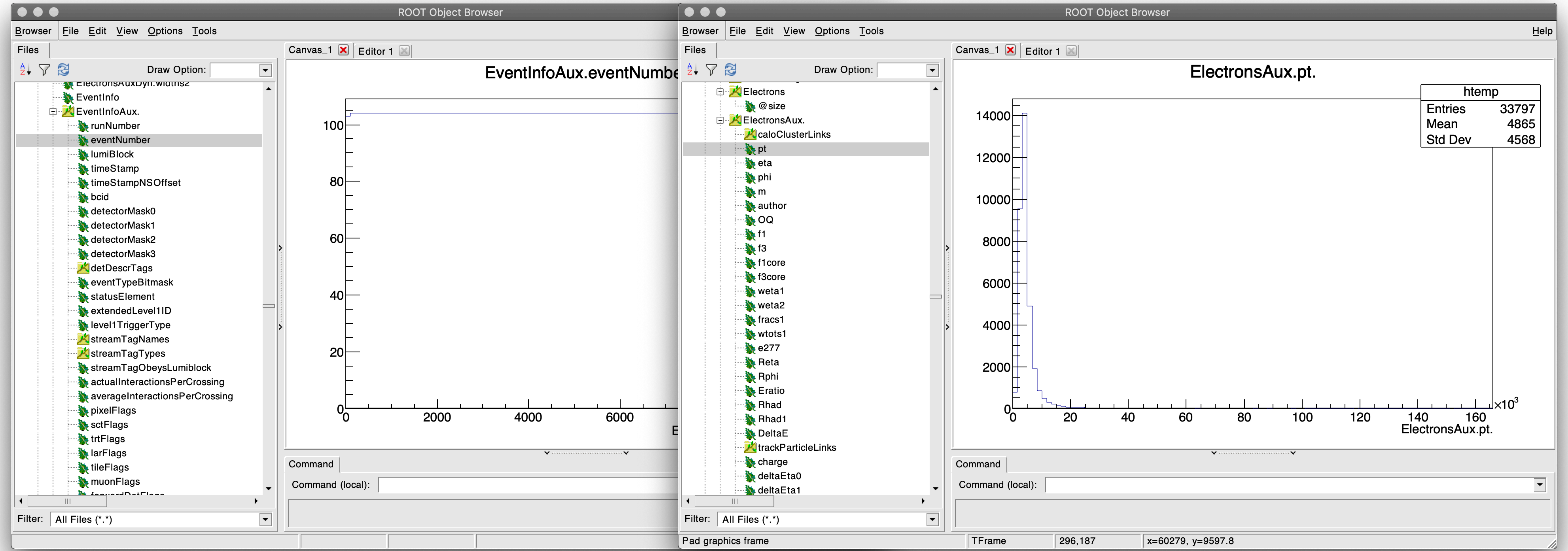
- Our code does not allow you to modify objects that you read in from an input file
  - This is to prevent hard-to-understand issues arising from a piece of code “silently” modifying an object that you did not intend
- So to perform object calibrations, you always have to make a copy in-memory of the objects that come from the input file
- But making copies of (especially large) xAOD containers is “expensive”
  - Creating / copying O(100) variables for possibly O(100) objects needs a lot of memory operations
- This is where “shallow copies” come to help
  - A shallow copied container does not hold any data itself when it is created. It just points back to the original container, and takes all variables from there.
  - Once you modify some variable in a shallow copy, the variable is created / copied from the original container, and becomes modifiable.
- You will see some technical details about this later on

xAOD Files



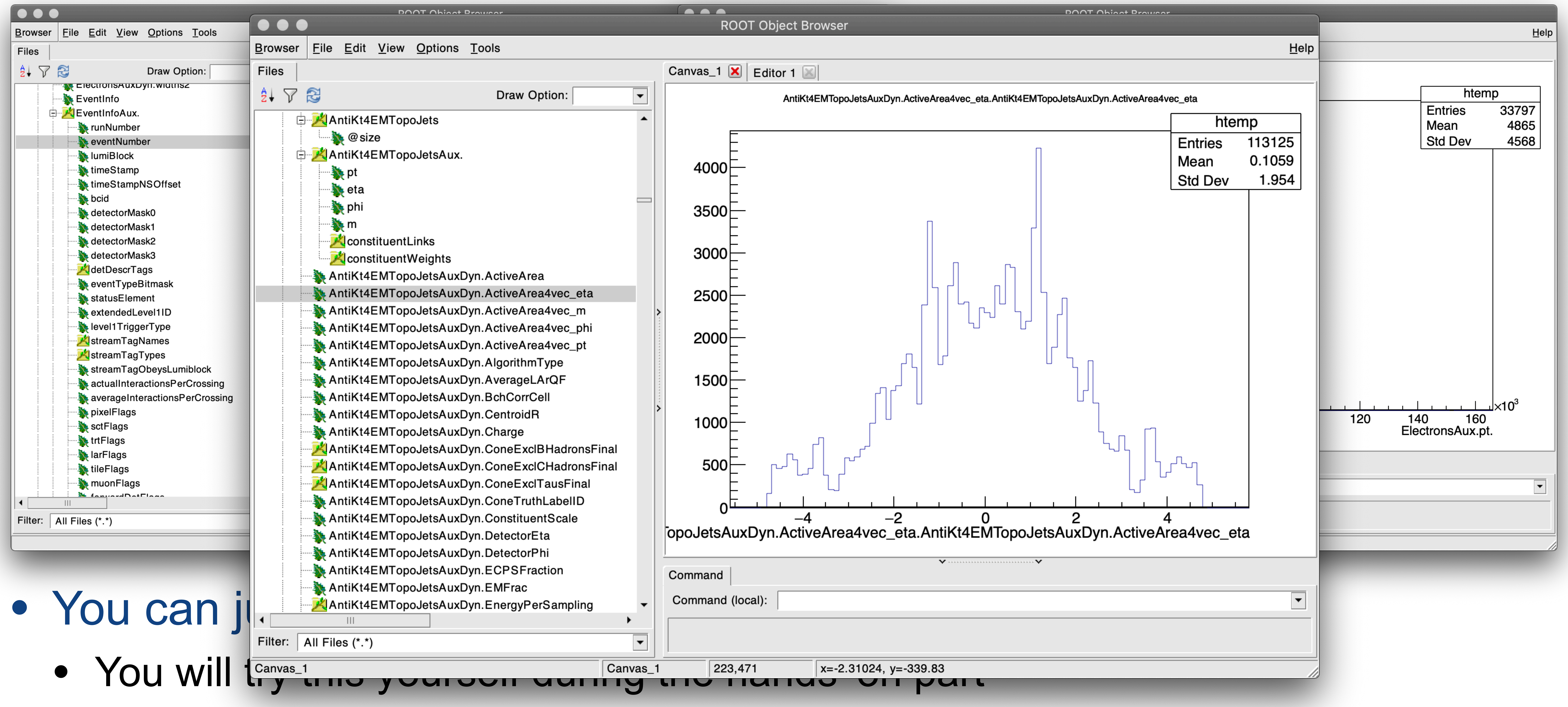
- You can just open (D)AOD files with ROOT directly
  - You will try this yourself during the hands' on part

xAOD Files



- You can just open (D)AOD files with ROOT directly
  - You will try this yourself during the hands' on part

xAOD Files



ROOT Object Browser

Browser File Edit View Options Tools Help

Files

Draw Option:

```
ElectronsAuxDyn.widths2
EventInfo
EventInfoAux.
  runNumber
  eventNumber
  lumiBlock
  timeStamp
  timeStampNSOffset
  bcid
  detectorMask0
  detectorMask1
  detectorMask2
  detectorMask3
  detDescrTags
  eventTypeBitmask
  statusElement
  extendedLevel1ID
  level1TriggerType
  streamTagNames
  streamTagTypes
  streamTagObeysLumiblock
  actualInteractionsPerCrossing
  averageInteractionsPerCrossing
  pixelFlags
  sctFlags
  trtFlags
  larFlags
  tileFlags
  muonFlags
  forwardDetFlags
```

Filter: All Files (*.*)

```
AntiKt4EMTopoJets
  @size
AntiKt4EMTopoJetsAux.
  pt
  eta
  phi
  m
  constituentLinks
  constituentWeights
AntiKt4EMTopoJetsAuxDyn.ActiveArea
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_m
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_phi
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_pt
AntiKt4EMTopoJetsAuxDyn.AlgorithmType
AntiKt4EMTopoJetsAuxDyn.AverageLArQF
AntiKt4EMTopoJetsAuxDyn.BchCorrCell
AntiKt4EMTopoJetsAuxDyn.CentroidR
AntiKt4EMTopoJetsAuxDyn.Charge
AntiKt4EMTopoJetsAuxDyn.ConeExclBHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclCHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclTausFinal
AntiKt4EMTopoJetsAuxDyn.ConeTruthLabelID
AntiKt4EMTopoJetsAuxDyn.ConstituentScale
AntiKt4EMTopoJetsAuxDyn.DetectorEta
AntiKt4EMTopoJetsAuxDyn.DetectorPhi
AntiKt4EMTopoJetsAuxDyn.ECPSFraction
AntiKt4EMTopoJetsAuxDyn.EMFrac
AntiKt4EMTopoJetsAuxDyn.EnergyPerSampling
```

Filter: All Files (*.*)

Canvas_1 Editor 1

AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta.AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta

htemp	
Entries	113125
Mean	0.1059
Std Dev	1.954

htemp	
Entries	33797
Mean	4865
Std Dev	4568

Command

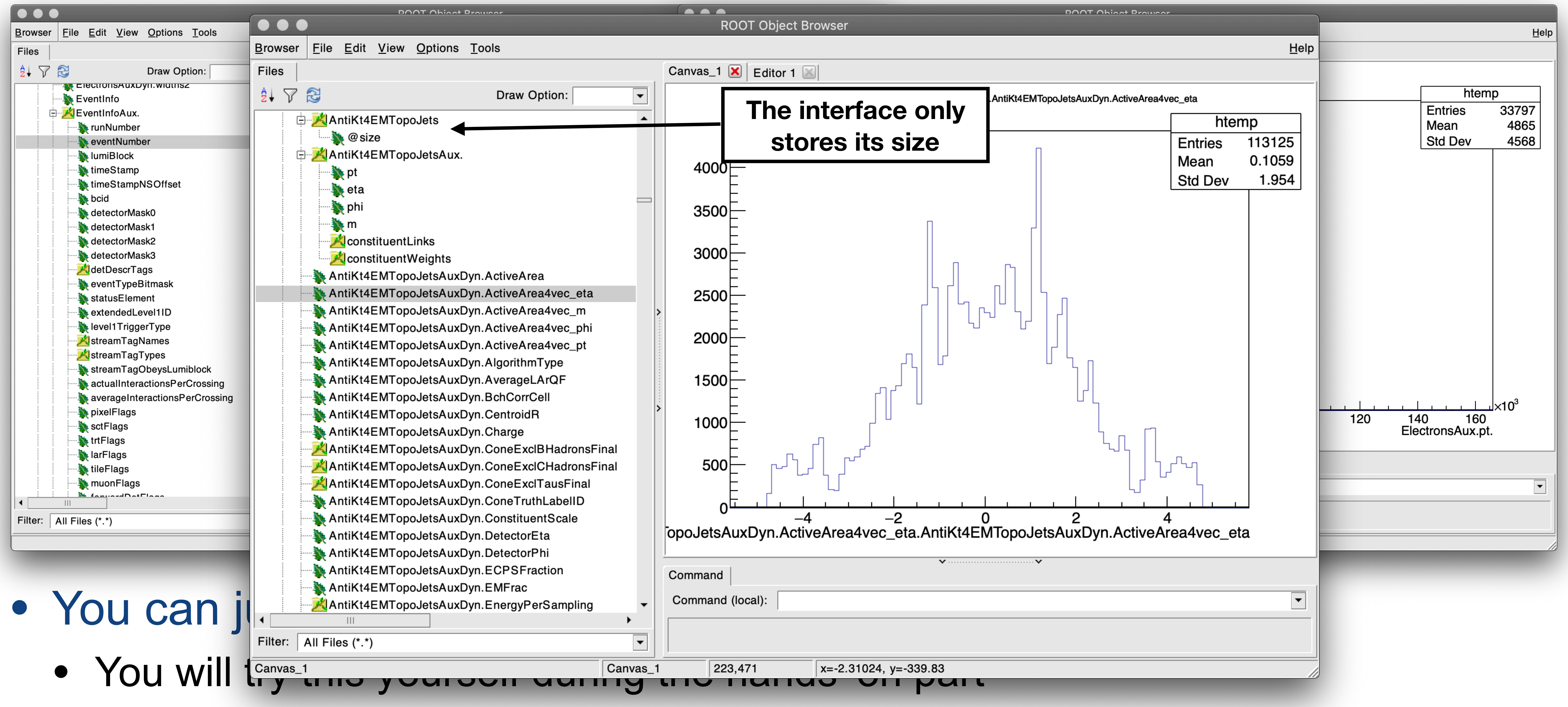
Command (local):

Canvas_1 | Canvas_1 | 223,471 | x=-2.31024, y=-339.83

4000
3500
3000
2500
2000
1500
1000
500
0
−4
−2
0
2
4
opoJetsAuxDyn.ActiveArea4vec_eta.AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
120
140
160
×10³
ElectronsAux.pt.

- You can j
  - You will try this yourself during the hands-on part

xAOD Files



ElectronsAuxDyn.widths2
EventInfo
EventInfoAux.
runNumber
eventNumber
lumiBlock
timeStamp
timeStampNSOffset
bcid
detectorMask0
detectorMask1
detectorMask2
detectorMask3
detDescrTags
eventTypeBitmask
statusElement
extendedLevel1ID
level1TriggerType
streamTagNames
streamTagTypes
streamTagObeysLumiblock
actualInteractionsPerCrossing
averageInteractionsPerCrossing
pixelFlags
sctFlags
trtFlags
larFlags
tileFlags
muonFlags
forwardDetFlags

Filter: All Files (*.*)

AntiKt4EMTopoJets
@size
AntiKt4EMTopoJetsAux.
pt
eta
phi
m
constituentLinks
constituentWeights
AntiKt4EMTopoJetsAuxDyn.ActiveArea
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_m
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_phi
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_pt
AntiKt4EMTopoJetsAuxDyn.AlgorithmType
AntiKt4EMTopoJetsAuxDyn.AverageLArQF
AntiKt4EMTopoJetsAuxDyn.BchCorrCell
AntiKt4EMTopoJetsAuxDyn.CentroidR
AntiKt4EMTopoJetsAuxDyn.Charge
AntiKt4EMTopoJetsAuxDyn.ConeExclBHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclCHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclTausFinal
AntiKt4EMTopoJetsAuxDyn.ConeTruthLabelID
AntiKt4EMTopoJetsAuxDyn.ConstituentScale
AntiKt4EMTopoJetsAuxDyn.DetectorEta
AntiKt4EMTopoJetsAuxDyn.DetectorPhi
AntiKt4EMTopoJetsAuxDyn.ECPSFraction
AntiKt4EMTopoJetsAuxDyn.EMFrac
AntiKt4EMTopoJetsAuxDyn.EnergyPerSampling

Filter: All Files (*.*)

The interface only stores its size

AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
4000
3500
3000
2500
2000
1500
1000
500
0
−4
−2
0
2
4
opoJetsAuxDyn.ActiveArea4vec_eta.AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
120
140
160
×10^3
ElectronsAux.pt.

htemp	
Entries	113125
Mean	0.1059
Std Dev	1.954

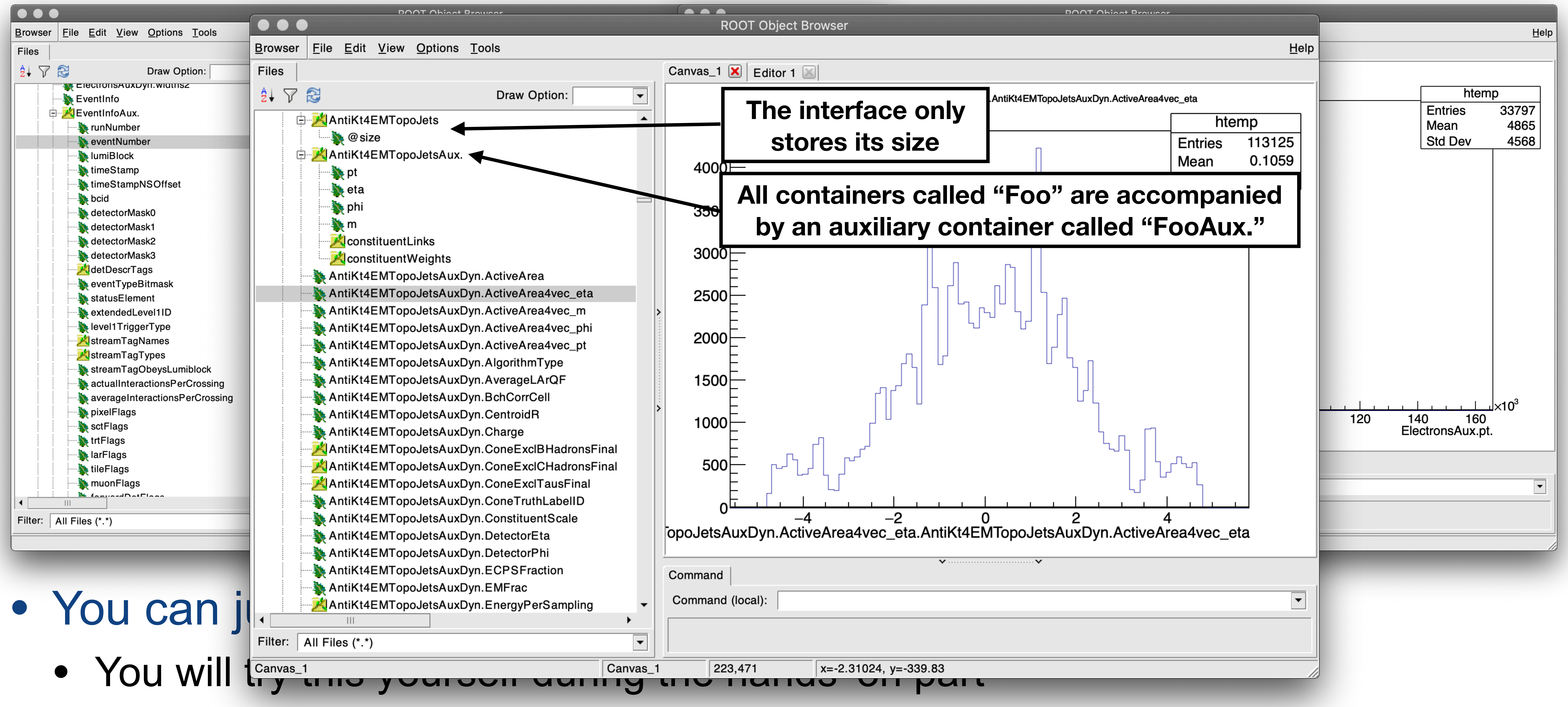
htemp	
Entries	33797
Mean	4865
Std Dev	4568

Command
Command (local):

Canvas_1 | Canvas_1 | 223,471 | x=-2.31024, y=-339.83

- You can j
  - You will try this yourself during the hands-on part

xAOD Files

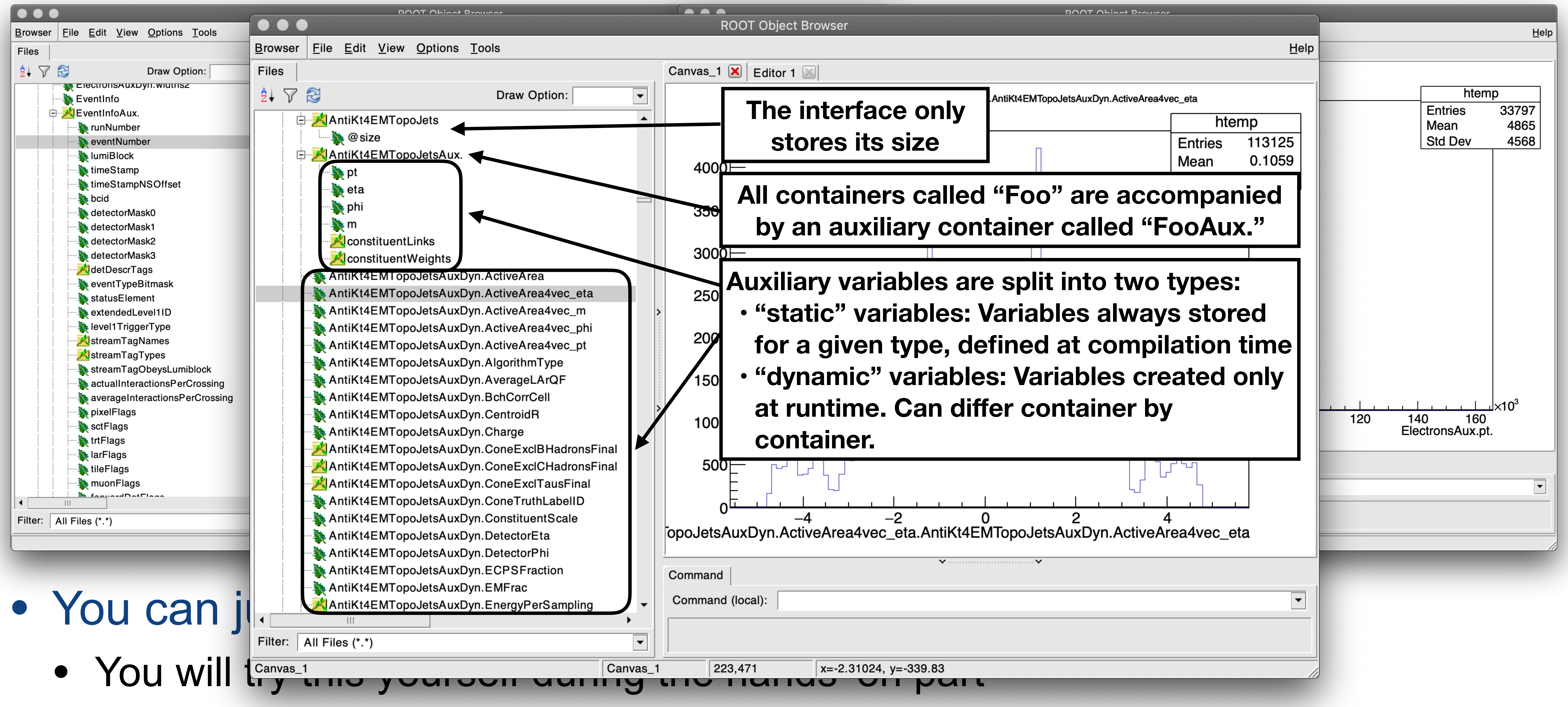

ROOT Object Browser
Browser File Edit View Options Tools Help
Files
Draw Option:
ElectronsAuxDyn.widths2
EventInfo
EventInfoAux.
runNumber
eventNumber
lumiBlock
timeStamp
timeStampNSOffset
bcid
detectorMask0
detectorMask1
detectorMask2
detectorMask3
detDescrTags
eventTypeBitmask
statusElement
extendedLevel1ID
level1TriggerType
streamTagNames
streamTagTypes
streamTagObeysLumiblock
actualInteractionsPerCrossing
averageInteractionsPerCrossing
pixelFlags
sctFlags
trtFlags
larFlags
tileFlags
muonFlags
Filter: All Files (*.*)
Canvas_1 Editor 1
AntiKt4EMTopoJets
@size
AntiKt4EMTopoJetsAux.
pt
eta
phi
m
constituentLinks
constituentWeights
AntiKt4EMTopoJetsAuxDyn.ActiveArea
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_m
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_phi
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_pt
AntiKt4EMTopoJetsAuxDyn.AlgorithmType
AntiKt4EMTopoJetsAuxDyn.AverageLArQF
AntiKt4EMTopoJetsAuxDyn.BchCorrCell
AntiKt4EMTopoJetsAuxDyn.CentroidR
AntiKt4EMTopoJetsAuxDyn.Charge
AntiKt4EMTopoJetsAuxDyn.ConeExclBHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclCHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclTausFinal
AntiKt4EMTopoJetsAuxDyn.ConeTruthLabelID
AntiKt4EMTopoJetsAuxDyn.ConstituentScale
AntiKt4EMTopoJetsAuxDyn.DetectorEta
AntiKt4EMTopoJetsAuxDyn.DetectorPhi
AntiKt4EMTopoJetsAuxDyn.ECPSFraction
AntiKt4EMTopoJetsAuxDyn.EMFrac
AntiKt4EMTopoJetsAuxDyn.EnergyPerSampling
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
htemp
Entries 113125
Mean 0.1059
4000
3500
3000
2500
2000
1500
1000
500
0
−4
−2
0
2
4
opoJetsAuxDyn.ActiveArea4vec_eta.AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
htemp
Entries 33797
Mean 4865
Std Dev 4568
120
140
160
×10³
ElectronsAux.pt.
Command
Command (local):
Canvas_1
Canvas_1
223,471
x=-2.31024, y=-339.83

The interface only stores its size

All containers called “Foo” are accompanied by an auxiliary container called “FooAux.”

- You can j
    - You will try this yourself during the hands-on part

xAOD Files


ROOT Object Browser
Browser File Edit View Options Tools Help
Files
Draw Option:
ElectronsAuxDyn.widths2
EventInfo
EventInfoAux.
runNumber
eventNumber
lumiBlock
timeStamp
timeStampNSOffset
bcid
detectorMask0
detectorMask1
detectorMask2
detectorMask3
detDescrTags
eventTypeBitmask
statusElement
extendedLevel1ID
level1TriggerType
streamTagNames
streamTagTypes
streamTagObeysLumiblock
actualInteractionsPerCrossing
averageInteractionsPerCrossing
pixelFlags
sctFlags
trtFlags
larFlags
tileFlags
muonFlags
forwardDetFlags
Filter: All Files (*.*)
AntiKt4EMTopoJets
@size
AntiKt4EMTopoJetsAux.
pt
eta
phi
m
constituentLinks
constituentWeights
AntiKt4EMTopoJetsAuxDyn.ActiveArea
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_m
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_phi
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_pt
AntiKt4EMTopoJetsAuxDyn.AlgorithmType
AntiKt4EMTopoJetsAuxDyn.AverageLArQF
AntiKt4EMTopoJetsAuxDyn.BchCorrCell
AntiKt4EMTopoJetsAuxDyn.CentroidR
AntiKt4EMTopoJetsAuxDyn.Charge
AntiKt4EMTopoJetsAuxDyn.ConeExclBHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclCHadronsFinal
AntiKt4EMTopoJetsAuxDyn.ConeExclTausFinal
AntiKt4EMTopoJetsAuxDyn.ConeTruthLabelID
AntiKt4EMTopoJetsAuxDyn.ConstituentScale
AntiKt4EMTopoJetsAuxDyn.DetectorEta
AntiKt4EMTopoJetsAuxDyn.DetectorPhi
AntiKt4EMTopoJetsAuxDyn.ECPSFraction
AntiKt4EMTopoJetsAuxDyn.EMFrac
AntiKt4EMTopoJetsAuxDyn.EnergyPerSampling
Filter: All Files (*.*)
Canvas_1 Editor 1
AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
htemp
Entries 113125
Mean 0.1059
4000
350
3000
250
200
150
100
500
0
−4 −2 0 2 4
opoJetsAuxDyn.ActiveArea4vec_eta.AntiKt4EMTopoJetsAuxDyn.ActiveArea4vec_eta
htemp
Entries 33797
Mean 4865
Std Dev 4568
120 140 160
×10³
ElectronsAux.pt.
Command
Command (local):
Canvas_1 Canvas_1 223,471 x=-2.31024, y=-339.83

The interface only stores its size

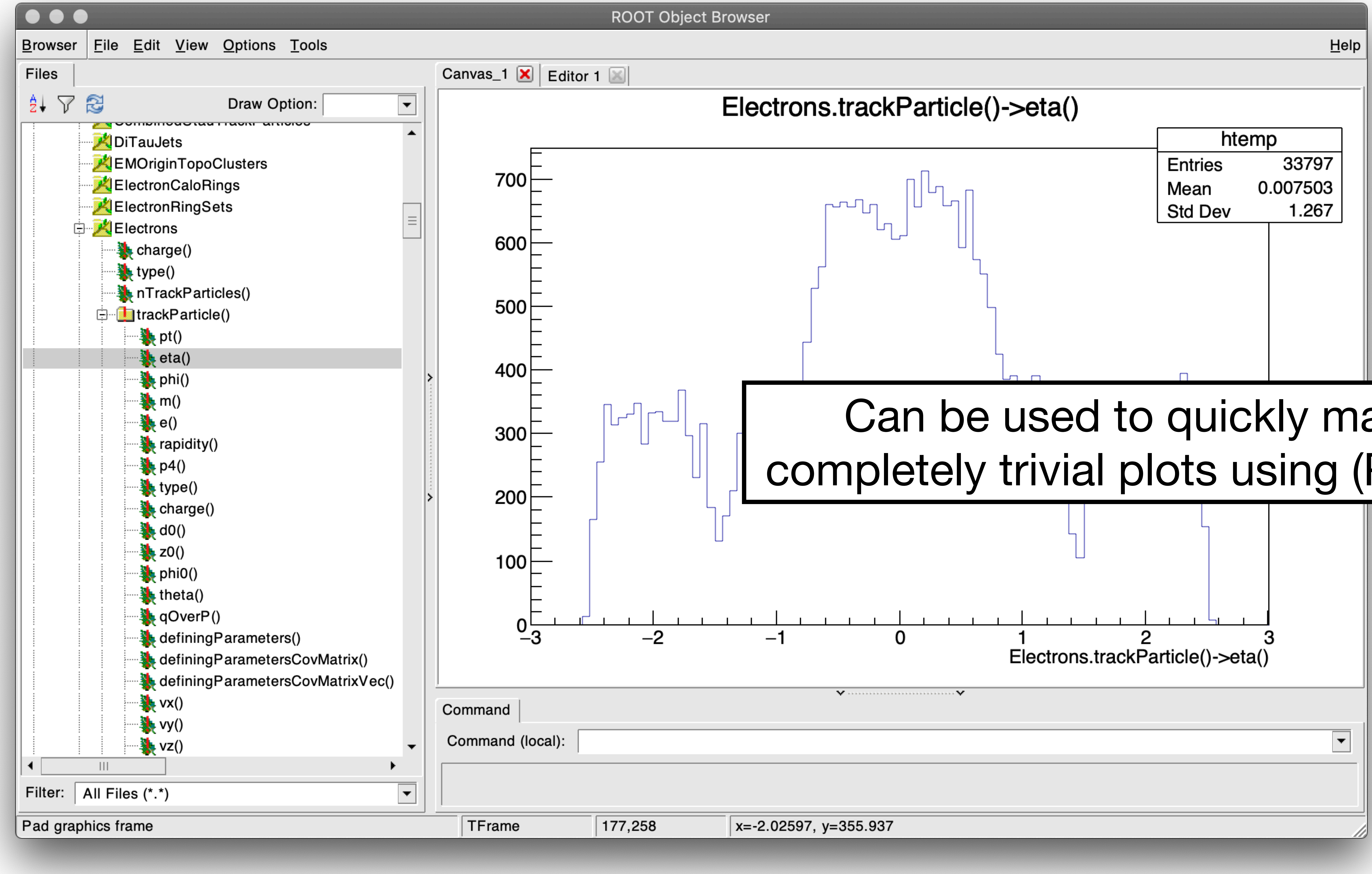
All containers called “Foo” are accompanied by an auxiliary container called “FooAux.”

Auxiliary variables are split into two types:

- “static” variables: Variables always stored for a given type, defined at compilation time
- “dynamic” variables: Variables created only at runtime. Can differ container by container.

- You can j
  - You will try this yourself during the hands-on part

xAOD Transient Trees



Can be used to quickly make some not completely trivial plots using (Py)ROOT scripts.

Summary

- Types of objects in the event
  - Interface objects (`xAOD::EventInfo`, `xAOD::Muon`, etc.): The objects that you should interact with. They (should) provide a convenient user interface.
  - Container objects (`xAOD::MuonContainer`, etc.): Containers of interface objects for types that have a variable number of objects event by event.
  - Auxiliary store objects (`xAOD::MuonAuxContainer`, etc.): Technical objects providing data storage for the interface objects/containers behind the scenes. Should only be accessed directly in “expert level” code.
- The rest of the EDM session will teach you about all the concrete types of xAOD objects that you may need to use during your analysis.

CERN

http://home.cern