Table of Contents

9.3.3 AssociationVector Container

Complete:
Detailed Review status

Contents

Introduction

`AssociationVector<KeyRefProd, CVal>` stores internally:

- a container of type `V` storing the associated quantities. Could be as simple as a `std::vector<float>`.
- a reference to a key collection (typically, and `edm::RefProd<...>`)

The container interface enforces that the stored container (`Cval`) has the same size as the associated collection (`KeyRefProd`).

`AssociationVector` can be used as lighter alternative to `AssociationMap<OneToValue<...> >` if every object in a collection has an associated quantity that can be stored *by value* in the association container.

AssociationVector Interface

`AssociationVector<KeyRefProd, CVal>` assumes there is a collection of objects of type `Key` already stored in the event, to which a reference object of type `KeyRefProd` refers to, and stores internally a collection `CVal` containing the same number of entries as the first collection.

`AssociationVector<KeyRefProd, CVal>` has an interface very similar to `std::vector<std::pair<Key, Val> >`, where `Val` is the object type contained in `CVal`, `Key` is a reference to an object in the collection referred to by `KeyRefProd`.

An example of usage of `AssociationVector` is the following:

```
typedef AssociationVector<MuonRefProd, vector<double> > MuonIsolationCollection;

Handle<MuonCollection> muons;
event.getByLabel( "muons", muons );

// create and fill the association vector
MuonIsolationCollection isolations( MuonRefProd( muons ) );
for( size_t i = 0; i < muons->size(); ++ i ) {
  isolations[ i ] = ....;
}

// read the association vector
for( size_t i = 0; i < isolations.size(); ++ i ) {
  MuonIsolationCollection::value_type iso = isolations( i );
  MuonRef mu = iso.first;
  double iso = iso.second;
```

```
}
```

AssociationVector Interactive Access

Given the simple internal structure of AssociationVector, interactive access is rather simple. The above example, modeling muon isolation, can be plotted against muon momentum as follows:

```
Events.Draw( "isolations.data_:globalMuons.pt()" );
```

Dictionary Generation

`AssociationVector<KeyRefProd, CVal>` requires the dictionary generation of both:

- `KeyRefProd`, and
- `CVal`
- `std::pair<Key, Val>`

where `Key` is a reference to an object in the collection referred to by `KeyRefProd`.

Moreover, the dictionary of the class itself has to be generated declaring the fields `transientVector_` ~~and `fixed_`~~ (Note: from CMSSW 7 the class template doesn't declare a fixed_ member anymore) as transient data members. Of course, the dictionary of `edm::Wrapper<edm::AssociationVector<...> >` has to be generated as well.

As example for the type:

- `edm::AssociationVector<edm::RefProd<std::vector<reco::Muon> >, std::vector<float> >"`

The following dictionary directives are needed:

```
<class name="edm::AssociationVector<edm::RefProd<std::vector<reco::Muon> >,
                                    std::vector<float>,
                                    edm::Ref<std::vector<reco::Muon>,
                                             reco::Muon,
                  edm::refhelper::FindUsingAdvance<std::vector<reco::Muon> >,
                                    reco::Muon> >,
                                    unsigned int>">
  <field name="transientVector_" transient="true"/>
</class>
<class name="std::pair<edm::Ref<std::vector<reco::Muon>,
                     reco::Muon,
                     edm::refhelper::FindUsingAdvance<std::vector<reco::Muon> >,
                                    reco::Muon>,
                                    float>"/>
<class name="std::vector<std::pair<edm::Ref<std::vector<reco::Muon>,
                     reco::Muon,
                     edm::refhelper::FindUsingAdvance<std::vector<reco::Muon> >,
                                          reco::Muon>,
                                           float> >"/>
<class name="edm::Wrapper<edm::AssociationVector<edm::RefProd<std::vector<reco::Muon> >,
                     std::vector<float>,
                     edm::Ref<std::vector<reco::Muon>,
                              reco::Muon,
                              edm::refhelper::FindUsingAdvance<std::vector<reco::Muon> >,
                                    reco::Muon> >,
                                    unsigned int> >"/>
```

Review Status

Editor/Reviewer and date	Comments
LucaLista - 26 Apr 2007	Updated to release 1_5_0_pre1
LucaLista - 23 Apr 2007	Page author and page content last edited

Responsible: LucaLista
Last reviewed by: PetarMaksimovic 28 Feb 2008

This topic: CMSPublic > WorkBookAssociationVector
Topic revision: r17 - 2018-01-17 - LuigiCalligaris