Clue

The BEE event model library.

T. Glebe

Contents

1 Utility routines 5
1.1 PkgIni — *Initialization of BEE package* 5
1.2 ROOTini — *Class to check whether the ROOT system is initialized* 6

2 RsegC — *HERA-B component id* 8

3 RecoSegment — *Reconstructed track class* 11

4 RecoTrack — *reconstructed track class* 18

5 RecoVertex — *Reconstructed vertex* 22

6 VDS_Track — *Track in vertex detector* 26

7 RICH_Track — *Track in RICH detector* 27

8 OTR_SpacePoint — *Space point in outer tracker* 28

9 MuonTrack — *Track in MUON detector* 29

10 TrackClones — *Clone class* 30

11 EcalHit — *Photon candidates found by ECAL* 32
11.1 — TrackIf methods — 36

12 MC_Track — *Monte Carlo Track* 38

13 MC_Vertex — *Monte Carlo Vertex* 44

14 StreamObject — *write/read ROOT files* 47

15 RtEvent — *BEE event definition.* 48

16 Wire — *Target wire class* 53

17 Topp — *TOPP table entry* 55

18 GTopp — *Representation of a Topp table entry* 57

19 PID — *Definitions for Particle ID* 59

20 PrimaryVertex — *Primary Vertex class* 60

This page has been automatically generated with DOC++
DOC++ is © 1995 by Roland Wunderling
Malte Zöckler

21 AnalysisChain — *Class to manage execution of a series of AnalysisIf objects* 62

22 ClueTTree — *Modified ROOT TTree class* 63

Class Graph 64

DOC++ is © 1995 by Roland Wunderling
Malte Zöckler

Clue provides the classes which are needed for the event structure plus basic classes which are needed for analyzation. The project was started by C. Borgmeier and I. Siccama.

1

Utility routines

Names

— Global Utility Functions —

template <typename T> inline unsigned int
getIDX (ArtePointer<T> ptr)
calc ArteTable index from Arte-Pointer

template <typename T> inline ArtePointer <T>
getPTR (unsigned int idx)
get ArtePointer from ArteTable index

template <typename TN, typename TO> inline ArtePointer <TN>
getPTR (const ArtePointer<TO>& ptr)
calc Clue ArtePointer from Arte ArtePointer

template <typename TN, typename TO> inline ArtePointer <TN>
getPTR (const ArteTableIterator<TO>& iter)
calc Clue ArtePointer from Arte ArtePointer

1.1 class **PkgIni** *Initialization of BEE package* 5

extern const PkgIni
ClueSteer *Global CLUE object which holds XCVS information.*

1.2 class **ROOTini** *Class to check whether the ROOT system is initialized ...* 6

1.1

class **PkgIni**

Initialization of BEE package

Public Members

— Constructors —

PkgIni () *for ROOT backward compatibility*

PkgIni (const char* nam, const char* vers, const char* dat)

~**PkgIni** ()

— Access methods —

const char* const
version () const *get string with version number*

const char* const
name () const *get string with package name*

const char* const
date () const *get birthday & time of package version*

const bool
debug () *toggle debug flag*

bool **isDebug** () const *return debug flag*

— Expert functions —

std::ostream&
print (std::ostream& os) const
used by operator<<()

Class to hold the XCVS information of a BEE package. Each project should create a global PkgIni object which is initialized with the projects XCVS information.

Author: T. Glebe

1.2 class **ROOTini**

Class to check whether the ROOT system is initialized

Public Members

ROOTini ()

Class to check whether the ROOT system is initialized. If not, initialize it.

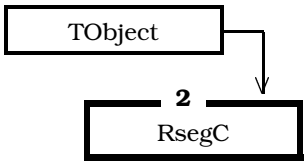
Author: T. Glebe

2

class **RsegC** : public TObject

HERA-B component id

Inheritance



Public Members

— Constructors —

RsegC ()

RsegC (const unsigned int rhs)

~**RsegC** ()

RsegC& **operator=** (const RsegC& rhs)

— Access methods —

unsigned int	**cmp** () const	*return bit pattern*
bool	**vxd** () const	*bit 0 (0x1): Vertex Detector*
bool	**patt** () const	*bit 1 (0x2): Pattern Tracker*
bool	**magt** () const	*bit 2 (0x4): Magnet Tracker*
bool	**muon** () const	*bit 3 (0x8): Muon Detector*
bool	**ecal** () const	*bit 4 (0x10): Ecal*
bool	**tar** () const	*bit 5 (0x20): Target Constraint*
bool	**trit** () const	*bit 6 (0x40): Trigger Tracker, betw RICH and ECAL*
bool	**bit7** () const	*bit 7 (0x80): not assigned yet*
bool	**slte** () const	*bit 8 (0x100): match SLT trigger tracks*

bool	**sltmu** () const	*bit 9 (0x200): match SLT trigger tracks*
bool	**bit10** () const	*bit 10 (0x400): not assigned yet*
bool	**bit11** () const	*bit 11 (0x800): not assigned yet*
bool	**bit12** () const	*bit 12 (0x1000): not assigned yet*
bool	**bit13** () const	*bit 13 (0x2000): not assigned yet*
bool	**bit14** () const	*bit 14 (0x4000): not assigned yet*
bool	**bit15** () const	*bit 15 (0x8000): not assigned yet*
bool	**bit16** () const	*bit 16 (0x10000): not assigned yet*
bool	**bit17** () const	*bit 17 (0x20000): not assigned yet*
bool	**bit18** () const	*bit 18 (0x40000): Match SVD-PC (marple RSEG only)*
bool	**bit19** () const	*bit 19 (0x80000): not assigned yet*
bool	**clonerm** () const	*bit 20 (0x100000): Used by Cloneremove.C*
bool	**bit21** () const	*bit 21 (0x200000): not assigned yet*
bool	**bit22** () const	*bit 22 (0x400000): not assigned yet*
bool	**bit23** () const	*bit 23 (0x800000): not assigned yet*
bool	**slt** () const	*bit 24 (0x1000000): SLT seeded segment*
bool	**trd** () const	*bit 25 (0x2000000): TRD straw device type*
bool	**rich** () const	*bit 26 (0x4000000): Cherenkov Counter device type*

bool	**ptc** () const	*bit 27 (0x8000000): HiPT Chamber device type*
bool	**msg** () const	*bit 28 (0x10000000): MSGC device type*
bool	**wch** () const	*bit 29 (0x20000000): honeycomb wire chamber device type*
bool	**sil** () const	*bit 30 (0x40000000): silicon strip device type*
bool	**bitsign** () const	*bit 31 (0x80000000): Kill bit, set if segment is deleted*

— Operators —

bool **operator==** (unsigned int rhs) const

bool **operator!=** (unsigned int rhs) const

const RsegC&
operator—= (const unsigned int rhs)

const RsegC&
operator&= (const unsigned int rhs)

— Expert functions —

std::ostream&
print (std::ostream& os) const

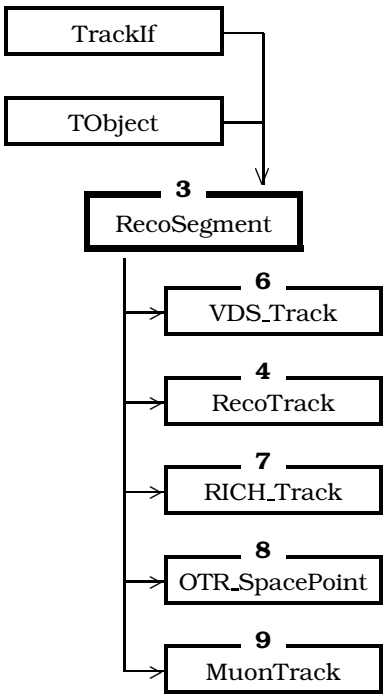
Author: T. Glebe

3

class **RecoSegment** : public TObject, public TrackIf

Reconstructed track class

Inheritance



Public Members

— Constructors —

RecoSegment ()

RecoSegment (const RecoSegment& rhs)

RecoSegment (const TrackIf& rhs)
store TrackIf

virtual ~**RecoSegment** ()

const RecoSegment&
operator= (const RecoSegment& rhs)

— ARTE only functions —

template <class T> void
convert (T rtra) *fill RecoSegment from RSEG or RTRA*

RecoSegment (const ArtePointer<RTRA>& r)
create object from RTRA table

RecoSegment (const ArtePointer<RSEG>& r)
create object from RSEG table

ArtePointer <RTRA>
rtra () const *return RTRA pointer*

void **set_rtra** (const ArtePointer<RTRA>& rtra)
set RTRA pointer

unsigned int
rtra () const *not available in ARTE*

— Operators —

bool **operator==** (const TrackIf& rhs) const
compare TrackIf pointers

bool **operator==** (const RecoSegment& rhs) const
compare Rtra value

bool **operator**< (const RecoSegment& rhs) const
compare Rtra value

— TrackIf methods —

float **x** () const x *of track*

float **y** () const y *of track*

float **x** (float z) const *linear extrapolation of* x *to z*

float **y** (float z) const *linear extrapolation of* y *to z*

float **z** () const z *of track*

float **zorig** () const *in case of propagation:* z *before propagation*

float **tx** () const dx/dz *slope of track* (t_x)

float **ty** () const dy/dz *slope of track* (t_y)

float **p** () const *momentum of track* (p)

float	**pt** () const	*transverse Track momentum* $p_t = p \cdot \sqrt{(t_x^2 + t_y^2)/(t_x^2 + t_y^2 + 1)}$
float	**pz** () const	*momentum along z:* $p_z = p/\sqrt{t_x^2 + t_y^2 + 1}$
float	**chi2** () const	χ^2 *of track fit*
unsigned short int	**ndf** () const	*degrees of freedom*
float	**phi** () const	*azimuthal angle* ϕ *[deg]*
float	**theta** () const	*polar angle* $\theta = \cos^{-1}(e_z)$ *[deg]*
float	**eta** () const	*rapidity* $\eta = -\log\tan(\theta/2.)$ *rapidity*
int	**charge** () const	*particle charge: -1 for neg. +1 for pos. & neutrals*
virtual float	**energy** (double mass = 0.) const	$E = \sqrt{m^2 + p^2}$
float	**xf** (double mass = 0.) const	*Feynman* x *variable:* $x_f = (p_z - E \cdot \sqrt{1 - \frac{2m_p}{920GeV/c + m_p}})/m_p$, m_p *= Proton mass*
float	**rap** (double mass = 0.) const	*Rapidity* $y = \frac{1}{2}\ln(\frac{E+p_z}{E-p_z})$
	— Access methods —	
float	**trprob** () const	*upper tail prob. of track* χ^2
void	**set_chi2** (const float chi2)	*set* χ^2
void	**set_nf** (const UShort_t nf)	*set degrees of freedom*
float	**pinv** () const	*inverse momentum of track*
float	**px** () const	$p_x = t_x \cdot p_z$
float	**py** () const	$p_y = t_y \cdot p_z$
RsegC	**cmp** () const	*Component id*
int	**isRICH** () const	*is RICH component id set?*
int	**isOTR** () const	*is OTR component id set?*

int	**isVDS** () const	*is VDS component id set?*
int	**isECAL** () const	*is ECAL component id set?*
void	**set_x** (const float x)	*set x parameter*
void	**set_y** (const float y)	*set y parameter*
void	**set_z** (const float z)	*set z parameter*
void	**set_tx** (const float tx)	*set t_x parameter*
void	**set_ty** (const float ty)	*set t_y parameter*
void	**set_p** (const float p)	*set momentum p*
float	**cov_x** (double dz=0.) const	*get $\sigma_x^2\|_{z+dz}$*
float	**cov_x_y** (double dz=0.) const	*get $\sigma_x \times \sigma_y\|_{z+dz}$*
float	**cov_y** (double dz=0.) const	*get $\sigma_y^2\|_{z+dz}$*
float	**cov_x_tx** (double dz=0.) const	*get $\sigma_x \times \sigma_{t_x}\|_{z+dz}$*
float	**cov_y_tx** (double dz=0.) const	*get $\sigma_y \times \sigma_{t_x}\|_{z+dz}$*
float	**cov_tx** () const	*get $\sigma_{t_x}^2$*
float	**cov_x_ty** (double dz=0.) const	*get $\sigma_x \times \sigma_{t_x}\|_{z+dz}$*
float	**cov_y_ty** (double dz=0.) const	*get $\sigma_y \times \sigma_{t_y}\|_{z+dz}$*
float	**cov_tx_ty** () const	*get $\sigma_{t_x} \times \sigma_{t_y}$*
float	**cov_ty** () const	*get $\sigma_{t_y}^2$*
float	**cov_x_p** (double dz=0.) const	*get $\sigma_x \times \sigma_p\|_{z+dz}$*
float	**cov_y_p** (double dz=0.) const	

		get $\sigma_y \times \sigma_p\|_{z+dz}$
float	**cov_tx_p** () const	*get* $\sigma_{t_x} \times \sigma_p$
float	**cov_ty_p** () const	*get* $\sigma_{t_y} \times \sigma_p$
float	**cov_p** () const	*get* σ_p^2
void	**set_cov_x** (const float c)	*set* σ_x^2
void	**set_cov_x_y** (const float c)	*set* $\sigma_x \times \sigma_y$
void	**set_cov_y** (const float c)	*set* σ_y^2
void	**set_cov_x_tx** (const float c)	*set* $\sigma_x \times \sigma_{t_x}$
void	**set_cov_y_tx** (const float c)	*set* $\sigma_y \times \sigma_{t_x}$
void	**set_cov_tx** (const float c)	*set* $\sigma_{t_x}^2$
void	**set_cov_x_ty** (const float c)	*set* $\sigma_x \times \sigma_{t_x}$
void	**set_cov_y_ty** (const float c)	*set* $\sigma_y \times \sigma_{t_y}$
void	**set_cov_tx_ty** (const float c)	*set* $\sigma_{t_x} \times \sigma_{t_y}$
void	**set_cov_ty** (const float c)	*set* $\sigma_{t_y}^2$
void	**set_cov_x_p** (const float c)	*set* $\sigma_x \times \sigma_p$
void	**set_cov_y_p** (const float c)	*set* $\sigma_y \times \sigma_p$
void	**set_cov_tx_p** (const float c)	*set* $\sigma_{t_x} \times \sigma_p$
void	**set_cov_ty_p** (const float c)	*set* $\sigma_{t_y} \times \sigma_p$
void	**set_cov_p** (const float c)	*set* σ_p^2
	— Expert functions —	
void	**collect** (vector<TrackIf*>& c) const	

collect pointers

bool **propagate** (const double zz)
propagate track to z

void **sPinvToP** () *transform cov$(Q/p) \rightarrow$ cov(p)*

void **sPToPinv** () *transform cov$(p) \rightarrow$ cov(Q/p)*

SVector <double, 3>
xvec () const $\vec{x} = (x,\ y,\ z)$

SVector <double, 3>
tvec () const $\vec{t} = (t_x,\ t_y,\ 1)$

SVector <double, 3>
evec () const $\vec{e} = (e_x,\ e_y,\ e_z)$ *unit vector along* $\vec{p}$

SVector <double, 3>
pvec () const $\vec{p} = (p_x,\ p_y,\ p_z)$ *momentum vector*

bool **FillCINV** () *fill COV_ matrix and compute inverse cov. matrix*

const SMatrix <double, 5> &
COV () const *return covariance matrix* ***Cov***

const SMatrix <double, 5> &
CINV () const *return inverse covariance matrix* $\boldsymbol{Cov}^{-1}$

virtual bool
isValid () const *is track valid?*

bool **PrepareForVertexing** ()
Prepare reconstructed track for vertexing with SVertex package: propagate track parameters and errors to $z = 0$, fill inverse covariance matrix

virtual void
valid () *mark track as valid*

virtual void
invalid () *mark track as invalid*

std::ostream&
print (std::ostream&) const
used by operator<<()

std::ostream&

print_cov (std::ostream&) const
print covariance matrix

— GRelation relations —

GRelationList <SPatternMatch>
pmatch *RecoTrack* $\rightarrow$ *PatternMatch*

const GRelationList <SPatternMatch> &
Pmatch () const

GRelationList <SPatternMatch> &
Pmatch ()

GRelationList <SPatternMatch> &
container (const SPatternMatch& p)

void **Print** (Option_t *option="")
only available in ROOT

Reconstructed track class. In contrast to ARTE, the momentum covariance matrix elements are computed with the momentum p and not with Q/p. The reason is that they enter in this way into the vertexing (Vt based vertexing codes). The transformation $cov_{ij}(Q/p) \rightarrow cov_{ij}(p)$ is done before writing the BEE files in the ARTE project cluearte. The RecoSegment memberfunctions sPinvToP() and sPtoPinv() do the appropriate transformation in the respective direction.

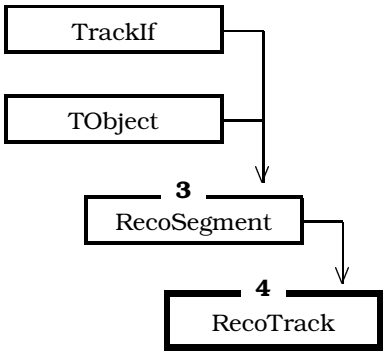
Author: C. Borgmeier, T. Glebe

4

class **RecoTrack** : public RecoSegment

reconstructed track class

Inheritance



Public Members

— Constructors —

RecoTrack ()

RecoTrack (const RecoTrack& rt)

~**RecoTrack** ()

RecoTrack (const ArtePointer<RTRA>& rtra,
double mass = 0)
not available in ROOT

RecoTrack (const ArtePointer<RSEG>& rseg)
not available in ROOT

const RecoTrack&
operator= (const RecoTrack& rhs)

— Access methods —

int	**pid** () const	*particle identification flag*
void	**set_pid** (int pid)	*set particle identification flag*
float	**le** () const	*global likelihood of electron hypothesis*

float	**lmu** () const	*global likelihood of muon hypothesis*
float	**lpi** () const	*global likelihood of pion hypothesis*
float	**lk** () const	*global likelihood of kaon hypothesis*
float	**lp** () const	*global likelihood of proton hypothesis*
int	**hsi** () const	*number of hits in the SVD*
int	**hin** () const	*number of hits in the inner tracker*
int	**hou** () const	*number of hits in the outer tracker*
int	**hri** () const	*number of hits in the RICH*
int	**hca** () const	*number of hits in the calorimeter*
int	**htr** () const	*number of hits in the TRD*
int	**hmu** () const	*number of hits in the muon chambers*
int	**fit** () const	*fit flag*
float	**mass** () const	*particle mass* m
void	**set_mass** (const float newMass)	*set particle mass* m
float	**energy** (double mass = 0.) const	$E = \sqrt{m^2 + p^2}$
float	**ebrems** () const	*possible Bremsstrahlung correction*
void	**set_ebrems** (const float ener)	*set bremsstrahlungs energy*
void	**ebrems_recovery** ()	*perform bremsstrahlungs recovery*
float	**cer** () const	*Cerenkov angle of associated RICH ring*
int	**mommask** () const	

Momentum determination flag:
1: estimated by REFIT on all hits assigned,
2: estimated by REFIT with rejection of hits in magnet,
3: estimated in VDS-ECAL match,
4: obtained during magnet tracking,
5: target constrain of OTR-PC segment,
6: PC-kick in the magnet is used for estimation,
7: estimated in VDS-RICH match,
8: estimated in VDS-PC match,
9: assigned by CATS for VDS segment

const ArtePointer <Topp>
topp () const *link to Topp table*

void **set_topp** (const ArtePointer<Topp>& topp)
set link to Topp table

— Expert functions —

std::ostream&
print (std::ostream& os) const
used by operator<<()

void **print_all** (std::ostream& os) const
print all available information

void **printMC** () const *print related MC tracks*

— CLUE relations —

ClueRelationList <MC_Track>
mtrack *RecoTrack ↔ MC_Track relation*

const ClueRelationList <MC_Track> &
mTrack () const

ClueRelationList <MC_Track> &
mTrack ()

ClueRelationList <MC_Track> &
container (const ArtePointer<MC_Track>& p)

ClueRelationList <TrackClones>

clones *RecoTrack ↔ TrackClones*

const ClueRelationList <TrackClones> &
Clones () const

ClueRelationList <TrackClones> &
Clones ()

ClueRelationList <TrackClones> &
container (const ArtePointer<TrackClones>& p)

ClueRelationList <RecoVertex>
rvertex *RecoTrack ↔ RecoVertex*

const ClueRelationList <RecoVertex> &
rVertex () const

ClueRelationList <RecoVertex> &
rVertex ()

ClueRelationList <RecoVertex> &
container (const ArtePointer<RecoVertex>& p)

ClueRelationList <EcalHit>
ehit *RecoTrack ↔ EcalHit*

const ClueRelationList <EcalHit> &
Ehit () const

ClueRelationList <EcalHit> &
Ehit ()

ClueRelationList <EcalHit> &
container (const ArtePointer<EcalHit>& p)

void **RelStreamer** (TBuffer& R__b,
const Version_t& StreamVersion)
only availabe in ROOT: call Streamer of relation container

void **Print** (Option_t* o="")
only available in ROOT

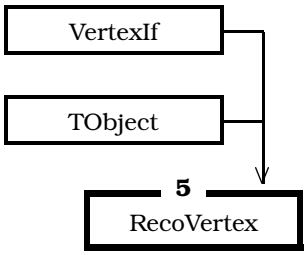
Author: C. Borgmeier, T. Glebe

5

class **RecoVertex** : public TObject, public VertexIf

Reconstructed vertex

Inheritance



Public Members

— Constructors —

RecoVertex () *default constructor*

RecoVertex (const RecoVertex& rhs)
copy constructor

~**RecoVertex** () *destructor*

RecoVertex (const ArtePointer<RVER>& rver)
from ArtePointer

const RecoVertex&
operator= (const RecoVertex& rhs)

— Operators —

bool **operator**< (const RecoVertex& rhs) const
$z_1 < z_2$

— VertexIf access methods —

float **vx** () const v_x = *vertex* x *position*

float **vy** () const v_y = *vertex* y *position*

float **vz** () const v_z = *vertex* z *position*

float **chi2** () const χ^2 *of vertex fit*

unsigned short int

ndf () const *degrees of freedom of vertex fit*

float **prob** () const *probability of the vertex fit*

unsigned short int
ntracks () const *no of tracks in vertex*

const TrackIf*
track (unsigned int i) const
returns pointer to track in vertex (slow access via relation!)

const TrackIf*&
track (unsigned int i)
returns null pointer + prints warning

float **vtx_cov_x** () const
σ_x^2 *of vertex*

float **vtx_cov_y** () const
σ_y^2 *of vertex*

float **vtx_cov_z** () const
σ_z^2 *of vertex*

— Access methods —

bool **valid** () const *is vertex valid?*

void **invalid** () *mark vertex as invalid*

void **set_valid** () *mark vertex as valid*

float **xerr** () const *error in x*

float **yerr** () const *error in y*

float **zerr** () const *error in z*

float **mass** () const *invariant mass in vertex*

SVector <double, 3>
vpos () const *vertex position $\vec{v} = (v_x,\ v_y,\ v_z)$*

const SMatrix <double, 3> &
VCOV () const *vertex covariance matrix*

std::ostream&
print (std::ostream& os) const
used by operator<<()

std::ostream&

print_cov (std::ostream& os) const
print covariance matrix

void **set_chi2** (float chi)
set vertex χ^2

void **set_ndf** (int nd) *set degrees of freedom*

void **set_mass** (float ms)
set invariant mass

float **vtx_cov_x_y** () const
$\sigma_x \times \sigma_y$ *of vertex*

float **vtx_cov_x_z** () const
$\sigma_x \times \sigma_z$ *of vertex*

float **vtx_cov_y_z** () const
$\sigma_y \times \sigma_z$ *of vertex*

void **set_vtx_cov_x** (const float c)
set σ_x^2 of vertex

void **set_vtx_cov_x_y** (const float c)
set $\sigma_x \times \sigma_y$ of vertex

void **set_vtx_cov_y** (const float c)
set σ_y^2 of vertex

void **set_vtx_cov_x_z** (const float c)
set $\sigma_x \times \sigma_z$ of vertex

void **set_vtx_cov_y_z** (const float c)
set $\sigma_y \times \sigma_z$ of vertex

void **set_vtx_cov_z** (const float c)
set σ_z^2 of vertex

void **set_xyz** (const float x, const float y,
const float z)
set vertex position

— CLUE relations —

ClueRelationList <RecoTrack>
out *RecoVertex $\leftrightarrow$ RecoTrack (relations to outgoing tracks)*

const ClueRelationList <RecoTrack> &
Out () const

ClueRelationList <RecoTrack> &
Out ()

ClueRelationList <RecoTrack> &

container (const ArtePointer<RecoTrack>& p)

void **RelStreamer** (TBuffer& R__b,
const Version_t& StreamVersion)
only availabe in ROOT: call Streamer of relation container

void **Clear** (Option_t *option="")
only available in ROOT

void **Print** (Option_t *option="")
only available in ROOT

void **Paint** (Option_t *option="")
only available in ROOT

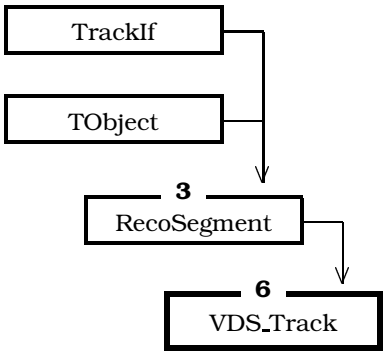
Author: C. Borgmeier, T. Glebe

6

class **VDS_Track** : public RecoSegment

Track in vertex detector

Inheritance



Public Members

VDS_Track ()

VDS_Track (const ArtePointer<RSEG>& rseg)

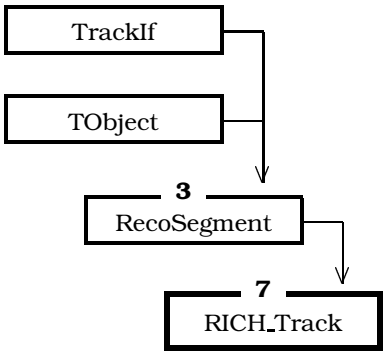
Author: C. Borgmeier, T. Glebe

7

class **RICH_Track** : public RecoSegment

Track in RICH detector

Inheritance



Public Members

	RICH_Track ()
	RICH_Track (const ArtePointer<RSEG>& rseg)
float	**radius**
float	**signalPhotons**
float	**backgndPhotons**
float	**fraction**

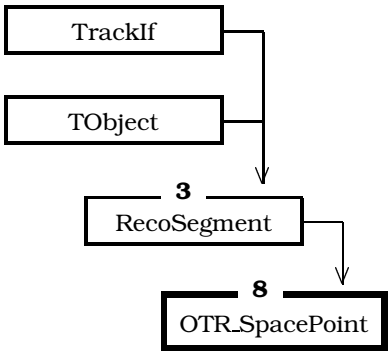
Author: C. Borgmeier, T. Glebe

8

class **OTR_SpacePoint** : public RecoSegment

Space point in outer tracker

Inheritance



Public Members

OTR_SpacePoint ()

OTR_SpacePoint (const ArtePointer<RSEG>& rseg)

int **nHits**

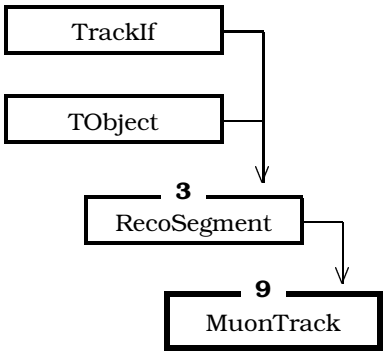
Author: C. Borgmeier, T. Glebe

9

class **MuonTrack** : public RecoSegment

Track in MUON detector

Inheritance



Public Members

MuonTrack ()

MuonTrack (const ArtePointer<RSEG>& rseg)

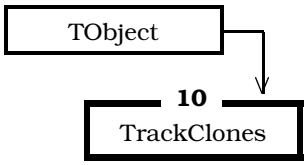
Author: C. Borgmeier, T. Glebe

10

class **TrackClones** : public TObject

Clone class

Inheritance



Public Members

— Constructors —

TrackClones ()

TrackClones (const TrackClones& rhs)

TrackClones (const RsegC& t)

~**TrackClones** ()

const TrackClones&
operator= (const TrackClones& rhs)

— Access methods —

RsegC **type** () const *type corresponds to rseg->cmp; type of incompatibility*

— CLUE relations —

ClueRelationList <RecoTrack>
rtrack *TrackClones ↔ RecoTrack*

const ClueRelationList <RecoTrack> &
rTrack () const

ClueRelationList <RecoTrack> &
rTrack ()

ClueRelationList <RecoTrack> &
container (const ArtePointer<RecoTrack>& p)

void **RelStreamer** (TBuffer& R__b,
const Version_t& StreamVersion)

only availabe in ROOT: call Streamer of relation container

Class for tracks sharing segments.

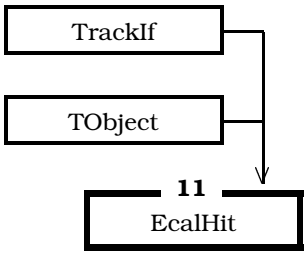
Author: C. Borgmeier, T. Glebe

11

class **EcalHit** : public TObject, public TrackIf

Photon candidates found by ECAL

Inheritance



Public Members

	— Constructors —	
	EcalHit ()	
	EcalHit (const ArtePointer<RCCL>& rccl)	*only available with ARTE*
	EcalHit (const EcalHit& rhs)	
	~**EcalHit** ()	
EcalHit&	**operator=** (const EcalHit& rhs)	
11.1	**— TrackIf methods —**	36
float	**x** () const	*x position of track*
float	**y** () const	*y position of track*
float	**z** () const	*z position of track*
float	**x** (float z) const	*Track x position at z*
float	**y** (float z) const	*Track y position at z*
float	**tx** () const	t_x *of cluster calculated from point* x_0
float	**ty** () const	t_y *of cluster calculated from point* x_0

float	**p** () const	*Photon track momentum* $\vert\vec{p}\vert = E_{cluster}$
float	**pt** () const	*transverse Photon track momentum:* $p_t = p \cdot \sqrt{(t_x^2 + t_y^2)/(t_x^2 + t_y^2 + 1)}$
float	**pz** () const	*Photon track momentum along z:* $p_z = p/\sqrt{t_x^2 + t_y^2 + 1}$
float	**phi** () const	*azimuthal angle* ϕ *[deg]*
float	**theta** () const	*polar angle* $\theta = \cos^{-1}(e_z)$ *[deg]*
float	**eta** () const	*pseudorapidity:* $\eta = -\log\tan(\theta/2.)$
int	**charge** () const	*Photon charge: 0*
float	**energy** (double rm = 0.) const	*Ecal cluster energy* $E_{cluster}$ *(rm argument has no effect)*
float	**xf** (double rm = 0.) const	*Feynman Variable:* $x_F = \frac{E+p_z}{(E+p_z)_{max}}$
float	**rap** (double rm = 0.) const	*Rapidity* $y = \frac{1}{2}\ln(\frac{E+p_z}{E-p_z})$
SVector <double, 3>	**xvec** () const	$\vec{x} = (x,\ y,\ z)$
SVector <double, 3>	**tvec** () const	$\vec{t} = (t_x,\ t_y,\ 1)$ *slope vector*
SVector <double, 3>	**evec** () const	$\vec{e} = (e_x,\ e_y,\ e_z)$ *unit vector along* $\vec{p}$
SVector <double, 3>	**pvec** () const	$\vec{p} = (p_x,\ p_y,\ p_z)$ *momentum vector*
bool	**isValid** () const	*is ECAL cluster valid?*
const SMatrix <double, 5> &	**COV** () const	*dummy function: return 0 matrix*
const SMatrix <double, 5> &	**CINV** () const	*dummy function: return 0 matrix*
float	**cov_x** (double dz=0.) const	

		get $\sigma_x^2\|_{z+dz}$
float	**cov_y** (double dz=0.) const	*get* $\sigma_y^2\|_{z+dz}$
float	**cov_tx** () const	*get* $\sigma_{t_x}^2 = \sigma_x^2/(z-z_0)^2$
float	**cov_ty** () const	*get* $\sigma_{t_y}^2 = \sigma_y^2/(z-z_0)^2$
float	**cov_p** () const	*get* σ_E^2
bool	**propagate** (const double dz)	*propagate track + errors*
void	**valid** ()	*mark cluster as valid*
void	**invalid** ()	*mark cluster as invalid*
void	**collect** (vector<TrackIf*>& c) const	*collect TrackIf pointers*
std::ostream&	**print** (std::ostream&) const	*used by operator<<()*
	— dummy functions —	
float	**chi2** () const	*dummy function: always return 0*
unsigned short int	**ndf** () const	*dummy function: always return 0*
	— EcalHit access functions —	
float	**zorig** () const	*return z before propagation*
float	**cov_x_tx** (double dz=0.) const	*get* $\sigma_x \times \sigma_{t_x}\|_{z+dz}$
float	**cov_y_ty** (double dz=0.) const	*get* $\sigma_y \times \sigma_{t_y}\|_{z+dz}$
	— Operators —	
bool	**operator==** (const TrackIf& rhs) const	*comparison between Track objects*
	— EcalHit Access methods —	
unsigned int	**width** () const	*width of cluster*
float	**x_c** () const	*x_c: cluster x*

float **y_c** () const — y_c: *cluster* y

float **z_c** () const — z_c: *cluster* z

float **px** () const — $t_x \cdot p_z$

float **py** () const — $t_y \cdot p_z$

float **Ez** () const — $E_z = E/\sqrt{t_x^2 + t_y^2 + 1}$

float **Et** () const — $E_t = E \cdot \sqrt{(t_x^2 + t_y^2)/(t_x^2 + t_y^2 + 1)}$

SVector <double, 3>
xc_vec () const — $\vec{x} = (x_c,\ y_c,\ z_c)$

SVector <double, 3>
Evec () const — $\vec{E} = (E_x,\ E_y,\ E_z)$

11.2 bool **calc_track** (const SVector<double, 6>& x0)
calculate track 36

11.3 bool **calc_track** (const VertexIf& rhs)
calculate track from Ecal cluster position and Vertex position, which is given as an argument 37

bool **calc_CINV** () — *calculate inverse cov. matrix*

bool **PrepareForVertexing** ()
Prepare reconstructed photon for vertexing with SVertex package: propagate track parameters and errors to $z = 0$*, fill inverse covariance matrix*

— CLUE relations —

ClueRelationList <RecoTrack>
rtrack — *EcalHit* $\leftrightarrow$ *RecoTrack*

const ClueRelationList <RecoTrack> &
rTrack () const

ClueRelationList <RecoTrack> &
rTrack ()

ClueRelationList <RecoTrack> &
container (const ArtePointer<RecoTrack>& p)

void **RelStreamer** (TBuffer& R__b,
const Version_t& StreamVersion)

only availabe in ROOT: call Streamer of relation container

— ROOT only functions —

void **Print** (Option_t * = "")

void **Paint** (Option_t *option)

ECAL Hit - Photon candidates

The errors of the cluster position at the ECAL are considered to be small and are not stored in this class. It is assumed for the moment that the major error comes from the unknown photon origin.

Author: T. Glebe

11.1 — TrackIf methods —

A photon track is defined by the ECAL cluster coordinates x_c, y_c, z_c and an origin x_0, y_0, z_0. Use the `calc_track()` function to define an origin.

11.2 bool **calc_track** (const SVector<double, 6>& x0)

calculate track

calculate track.

Calculate track parameters and track covariance matrix using the photon origin and the errors of the origin given by the vector x_0.

$$\vec{x}_0 = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ \sigma_{x_0} \\ \sigma_{y_0} \\ \sigma_{z_0} \end{pmatrix}$$

11.3

bool **calc_track** (const VertexIf& rhs)

calculate track from Ecal cluster position and Vertex position, which is given as an argument

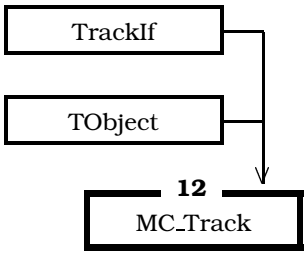
calculate track from Ecal cluster position and Vertex position, which is given as an argument. Use the Vertex covariance matrix to calculate the covariance matrix of the photon candidate.

12

class **MC_Track** : public TObject, public TrackIf

Monte Carlo Track

Inheritance



Public Members

— Constructors —

MC_Track ()

MC_Track (const MC_Track& rhs)

MC_Track (const ArtePointer<MTRA>& t)
constructor reading from ARTE table

~**MC_Track** ()

MC_Track&
operator= (const MC_Track&)

— TrackIf Access methods —

float	**x** () const	*Track x position*
float	**y** () const	*Track y position*
float	**x** (float z) const	*Track x position at z*
float	**y** (float z) const	*Track y position at z*
float	**z** () const	*Track z position*
float	**tx** () const	*Track slope t_x*
float	**ty** () const	*Track slope t_y*

float	**p** () const	*Track momentum* p
float	**pt** () const	*transverse Track momentum* p_t *(⊥ to z)*
float	**pz** () const	*Track momentum along* z
float	**chi2** () const	*dummy function: returns 0*
unsigned short int	**ndf** () const	*dummy function: returns 0*
float	**phi** () const	*azimuthal angle* ϕ *[deg]*
float	**theta** () const	*polar angle* $\theta = \cos^{-1}(e_z)$ *[deg]*
float	**eta** () const	*pseudorapidity:* $\eta = -\log\tan(\theta/2.)$
int	**charge** () const	*track charge*
float	**energy** (double rm = 0.) const	*track energy* $E = \sqrt{m^2+p^2}$, *rm is ignored*
float	**xf** (double rm = 0.) const	*Feynman Variable:* $x_F = \frac{E+p_z}{(E+p_z)_{max}}$, *rm is ignored*
float	**rap** (double rm = 0.) const	*Rapidity* $y = \frac{1}{2}\ln(\frac{E+p_z}{E-p_z})$, *rm is ignored*
SVector <double, 3>	**xvec** () const	$\vec{x} = (x,\ y,\ z)$
SVector <double, 3>	**tvec** () const	$\vec{t} = (t_x,\ t_y,\ 1)$ *slope vector*
SVector <double, 3>	**evec** () const	$\vec{e} = (e_x,\ e_y,\ e_z)$ *unit vector along* $\vec{p}$
SVector <double, 3>	**pvec** () const	$\vec{p} = (p_x,\ p_y,\ p_z)$ *momentum vector*
const SMatrix <double, 5> &	**COV** () const	*dummy function: returns zero matrix*
const SMatrix <double, 5> &		

	CINV () const	*dummy function: returns zero matrix*
float	**cov_x** (double dz=0.) const	*dummy function: return 0*
float	**cov_y** (double dz=0.) const	*dummy function: return 0*
float	**cov_tx** () const	*dummy function: return 0*
float	**cov_ty** () const	*dummy function: return 0*
float	**cov_p** () const	*dummy function: return 0*
bool	**isValid** () const	*dummy function: always return true*
	— Operators —	
bool	**operator==** (const TrackIf& rhs) const	*comparison between Track objects*
bool	**operator==** (const GTopp& rhs) const	*compare with GTOPP*
	— Expert methods —	
void	**valid** ()	*dummy function: do nothing*
void	**invalid** ()	*dummy function: do nothing*
void	**collect** (vector<TrackIf*>& c) const	*collect TrackIf pointers*
bool	**propagate** (const double dz)	*dummy function: do nothing*
std::ostream&	**print** (std::ostream&) const	*used by operator<<()*
	— Access methods —	
float	**mass** () const	*return topp mass* m
const char* const	**name** () const	*return topp name*
unsigned int		

flag () const *MC flag: 0: just created; not yet processed*
1: expected to decay according to QMTRA_LENG
2: decayed resonance or electromagnetic decay (production vertex = decay vertex)
3: decay with non-zero track length
4: track ends in a electromagnetic or hadronic interaction
5: track leaves the detector volume
6: track stops inside detector volume
7: track curls up inside detector volume
8: track stopped by user

const ToppIf&
topp () const *link to Topp table*

const ArtePointer <MC_Vertex>
bVertex () const *pointer to begin vertex*

void **set_bVertex** (const ArtePointer<MC_Vertex>& p)
set pointer to begin vertex

const ArtePointer <MC_Track>
mother () const *pointer to begin vertex*

void **set_mother** (const ArtePointer<MC_Track>& t)
set pointer to begin vertex

12.1 void **print_decay** (unsigned int level, char opt = ' ',
const ArtePointer<MC_Track>
mtra = 0,
unsigned int lvl=0) const
print MC decay chain 43

unsigned int
outsize () const *no of outgoing tracks*

— CLUE relations —

ClueRelationList <RecoTrack>

rtrack *MC_Track ↔ RecoTrack relation*

const ClueRelationList <RecoTrack> &
rTrack () const

ClueRelationList <RecoTrack> &
rTrack ()

ClueRelationList <RecoTrack> &
container (const ArtePointer<RecoTrack>& p)

ClueRelationList <MC_Vertex>
endVtx *MC_Track ↔ MC_Vertex relation (end vertices)*

const ClueRelationList <MC_Vertex> &
EndVtx () const

ClueRelationList <MC_Vertex> &
EndVtx ()

ClueRelationList <MC_Vertex> &
container (const ArtePointer<MC_Vertex>& p)

ClueRelationList <MC_Track>
endTrack *MC_Track ↔ MC_Track relation (outgoing tracks)*

const ClueRelationList <MC_Track> &
EndTrack () const

ClueRelationList <MC_Track> &
EndTrack ()

ClueRelationList <MC_Track> &
container (const ArtePointer<MC_Track>& p)

— ROOT only functions —

void **RelStreamer** (TBuffer& R__b,
const Version_t& StreamVersion)
call Streamer of relation container

void **Print** (Option_t *option="")

12.2 virtual void
Paint (Option_t *option="")
Do not call this function directly, use Draw() instead ... 43

Monte Carlo Track. In case of prompt decays (e.g. $\phi \rightarrow K^+ + K^-$) no begin- or end-vertices exist in ARTE (to reduce event size). Therefore we need in addition to begin- and end-vertex relations relations to mother- and outgoing tracks. In case of decays which have a travel distance this introduces some redundancy: the mother & outgoing tracks can be accessed either directly or via the begin- and end-vertices.

Author: C. Borgmeier, T. Glebe

12.1

void **print_decay** (unsigned int level, char opt = ' ', const ArtePointer<MC_Track> mtra = 0, unsigned int lvl=0) const

print MC decay chain

print MC decay chain. Options for additional information: f: print track flag m: print ArtePointer<MC_Track> r: print ArtePointer<RecoTrack> (matches to reconstructed tracks)

12.2

virtual void **Paint** (Option_t *option="")

Do not call this function directly, use Draw() instead

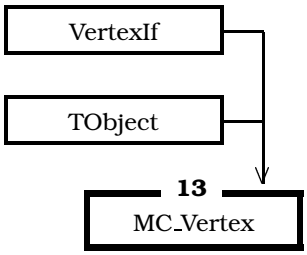
Do not call this function directly, use Draw() instead. Before you can draw, create a TView.

13

class **MC_Vertex** : public TObject, public VertexIf

Monte Carlo Vertex

Inheritance



Public Members

MC_Vertex ()

MC_Vertex (const MC_Vertex& rhs)

MC_Vertex (const ArtePointer<MVER>& mv)
only available in ARTE

~**MC_Vertex** ()

MC_Vertex&
operator= (const MC_Vertex&)

std::ostream&
print (std::ostream&) const
used by operator<<()

— VertexIf access methods —

float	**vx** () const	v_x = *vertex* x *position*
float	**vy** () const	v_y = *vertex* y *position*
float	**vz** () const	v_z = *vertex* z *position*
float	**chi2** () const	*dummy function: return 0*
unsigned short int	**ndf** () const	*dummy function: return 0*
float	**prob** () const	*dummy function: return 1*

unsigned short int
ntracks () const *no of tracks in vertex*

const TrackIf*
track (unsigned int i) const
returns pointer to track in vertex (slow access via relation!)

const TrackIf*&
track (unsigned int i)
returns null pointer + prints warning

SVector <double, 3>
vpos () const *vertex position* $\vec{v} = (v_x,\ v_y,\ v_z)$

const SMatrix <double, 3> &
VCOV () const *dummy function: return zero matrix*

float **vtx_cov_x** () const
dummy function: return 0

float **vtx_cov_y** () const
dummy function: return 0

float **vtx_cov_z** () const
dummy function: return 0

— Access methods —

const ArtePointer <MC_Track>
mother () const *return pointer to mother track*

void **set_mother** (const ArtePointer<MC_Track>& p)
set pointer to mother track

— CLUE relations —

ClueRelationList <MC_Track>
out *MC_Vertex* $\leftrightarrow$ *MC_Track (relations to outgoing tracks)*

const ClueRelationList <MC_Track> &
Out () const

ClueRelationList <MC_Track> &
Out ()

ClueRelationList <MC_Track> &
container (const ArtePointer<MC_Track>& p)

void **RelStreamer** (TBuffer& R__b,
const Version_t& StreamVersion)

DOC++ is © 1995 by Roland Wunderling
Malte Zöckler

only availabe in ROOT: call Streamer of relation container

void **Print** (Option_t *option="")
only available in ROOT

void **Paint** (Option_t *option="")
only available in ROOT

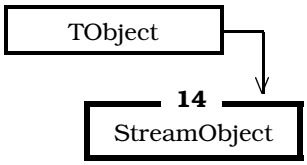
Author: C. Borgmeier, T. Glebe

14

class **StreamObject** : public TObject

write/read ROOT files

Inheritance



Public Members

StreamObject ()

StreamObject (RtEvent*)

Helper class to write and read CLUE events from/to ROOT files.

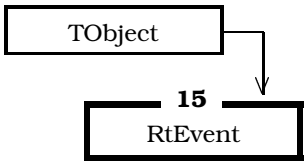
Author: C. Borgmeier, T. Glebe

15

class **RtEvent** : public TObject

BEE event definition.

Inheritance



Public Members

		— Constructors —		
		RtEvent ()		
	virtual	~**RtEvent** ()		
		— Event Header Information —		
	long int	**unixtime** () const	*seconds since 1.1.70 0:00 somewhere in the U.S.*	
	int	**evt** () const	*event number*	
	int	**run** () const	*run number*	
15.1	int	**exp** () const	*experiment number*	52
	float	**vfld** () const	*version of the mag field MC and reco*	
	int	**flagdst** () const	*event classification flag from/to DST*	
	int	**dbmajor** () const	*major key for constants on data base*	
	int	**dbminor** () const	*minor key for constants on data base*	
	int	**flag** () const	*event classification flag from/to EDIR if no EDIR: same as flagdst*	

	unsigned int	**bx** () const	*bx number from DIHD*	
15.2	int	**tcode** () const	*Trigger code*	52
	float	**ir** () const	*Interaction rate*	
	float	**bg** () const	*Background rate*	
	float	**rad0** () const	*Cherenkov radius for* $\beta = 1$	
	int	**hvds** () const	*Hits in VDS*	
	int	**hwch** () const	*Hits in wire chambers*	
	int	**hmsg** () const	*Hits in MSG microstrip chambers*	
	int	**hptc** () const	*Hits in PTC chambers (pi-pi trigger)*	
	int	**hrich** () const	*Hits in Cerenkov counter*	
	int	**htrd** () const	*Hits in forward TRD*	
	int	**hcal** () const	*Hits in electromagnetic calorimeter*	
	int	**hmuo** () const	*Hits in muon outer chambers*	
	int	**hmux** () const	*Hits in muon central chambers*	
	vector <float>	**user**	*user space for ROOT file*	
		— Container —		
	GClonesArray <Topp*> *	**topp** () const	*get Topp table pointer*	
	GClonesArray <RecoTrack*> *	**tracks** () const	*get RecoTrack table pointer*	
	GClonesArray <VDS_Track*> *	**vds** () const		
	GClonesArray <OTR_SpacePoint*> *	**otr** () const		
	GClonesArray <RICH_Track*> *	**rich** () const		
	GClonesArray <EcalHit*> *	**ecal** () const		
	GClonesArray <MuonTrack*> *			

muon () const

GClonesArray <RecoVertex*> *
vertices () const

GClonesArray <Wire*> *
wires () const

GClonesArray <TrackClones*> *
clones () const

GClonesArray <MC_Track*> *
mcTracks () const

GClonesArray <MC_Vertex*> *
mcVertices () const

— Event algorithms —

bool **PrepareRecoTracksForVertexing** ()
Prepare reconstructed tracks for vertexing with SVertex package: call `Prepare-ForVertexing()` *for each* `RecoTrack` *object in the* `Tracks` *container*

bool **PrepareEcalHitForVertexing** (const SVector<double, 6>& rhs)
Prepare all Ecal cluster for vertexing with SVertex package: calculate track parameters from given origin and call `PrepareForVertexing()` *for each* `EcalHit` *object in the* `Ecal` *container*

bool **PrepareEcalHitForVertexing** (const VertexIf& rhs)

		Prepare all Ecal cluster for vertexing with SVertex package: calculate track parameters from given vertex (which is used as the origin) and call `PrepareForVertexing()` *for each* `EcalHit` *object in the* `Ecal` *container*
	— Expert functions —	
std::ostream&	**print** (std::ostream& os) const	*used in operator<<()*
bool	**FillTopp** ()	*fill Topp table from ARTE TOPP*
bool	**FillHeader** ()	*fill event header from ARTE EVHD, DIHD*
void	**unlock** ()	*initialize objects contained in GClonesArrays*
	— ROOT only functions —	
void	**Clear** (Option_t * = "")	
void	**Delete** (Option_t * = "")	
void	**Print** (Option_t *options="")	
void	**StreamUser** (TBuffer &R__b)	*read/write user field*
void	**Clear** ()	*replacement of ROOT Clear()*
void	**Delete** ()	*replacement of ROOT Delete()*

Holds information of ARTE in C++ objects. The `GClonesArray`-container can also be accessed via the ARTE table interface.

Author: C. Borgmeier, T. Glebe

15.1

int **exp** () const

experiment number

experiment number.
exp > 0: real data
exp < 0: Monte Carlo; specifies the lab:
-1: DESY HH
-2: DESY Zeuthen
...: not yet specified

15.2

int **tcode** () const

Trigger code

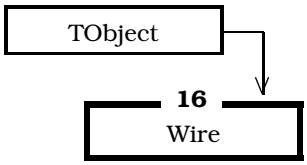
Trigger code.
0: (FLT) FLT trigger
2: (VME) Standalone trigger
4: (LEMO1) ECAL Internal LED calibration
5: (LEMO2) Front Panel trigger for test
6: (LEMO3) Front Panel trigger for test
7: (LEMO4) Front Panel trigger for test
8: FLT + RANDOM
9: (RANDOM) Random trigger
10: VME + RANDOM
12: LEMO1 + RANDOM
13: LEMO2 + RANDOM
14: LEMO3 + RANDOM
15: LEMO4 + RANDOM

16

class **Wire** : public TObject

Target wire class

Inheritance



Public Members

— Constructors —

Wire ()

Wire (const ArtePointer<GTAR>& wire)
only available in ARTE

Wire (const ArtePointer<RTAR>& wire)
only available in ARTE

Wire (const Wire& rhs)

~**Wire** ()

— Access methods —

float	**x** () const	*x of wire center*
float	**y** () const	*y of wire center*
float	**z** () const	*z of wire center*
float	**dx** () const	*size in x-direction*
float	**dy** () const	*size in y-direction*
float	**dz** () const	*size in z-direction*
SVector <double, 3>	**xvec** () const	*position vector* $\vec{x} = (x,\ y,\ z)$
SVector <double, 3>	**dxvec** () const	*size vector* $\vec{dx} = (dx,\ dy,\ dz)$
SVector <double, 3>		

dvec () const *direction vector* $\vec{d} = (1,\ 0,\ 0)$ *or* $\vec{d} = (0,\ 1,\ 0)$

— Expert functions —

std::ostream&
print (std::ostream&) const
used by operator<<()

— ROOT only functions —

void **Print** (Option_t *options="")

void **Paint** (Option_t *option="")

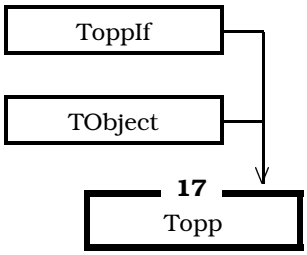
Author: T. Glebe

17

class **Topp** : public TObject, public ToppIf

TOPP table entry

Inheritance



Public Members

— Constructors —

Topp ()

Topp (const ArtePointer<TOPP>& topp)
only available with ARTE

const Topp&
operator= (const ArtePointer<TOPP>& topp)

— ToppIf Access methods —

const char* const	**name** () const	*particle name*
float	**mass** () const	*mass*
float	**wid** () const	*width*
float	**trbw** () const	*truncate Breit-Wigner at mass - trbw*
float	**life** () const	*life time*
int	**chrg** () const	*3 * charge*
bool	**valid** () const	*Do we represent a valid Topp entry?*

const ToppIf&

	anti () const	*get anti particle*
int	**lund** () const	*LUND code*
	— Access Methods —	
int	**colo** () const	*colour information: 0 = colour singlet; 2 = gluon; 1 = quark or anti-diquark; -1 = anti-quark or diquark*
int	**deca** () const	*LUND decay flag (0 = no decay)*
int	**gean** () const	*GEANT code*
const ArtePointer <Topp>	**anti_ptr** () const	*anti-particle row number*
void	**set_anti** (const ArtePointer<Topp>& topp)	*set anti-particle row number*
	— Operators —	
bool	**operator==** (const ToppIf& rhs) const	
	— Expert functions —	
std::ostream&	**print** (std::ostream&) const	*used by operator<<()*
	— ROOT only functions —	
void	**Print** (Option_t *option="")	

TOPP table entry.

Class to represent particle properties.

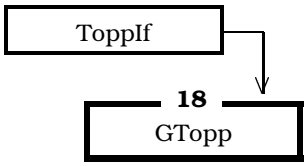
Author: C. Borgmeier, T. Glebe

18

class **GTopp** : public ToppIf

Representation of a Topp table entry

Inheritance



Public Members

— Constructors —

GTopp ()

GTopp (const GTopp& rhs)

GTopp (const char* name)

GTopp (const char* name, int chrg)

virtual ~**GTopp** ()

— ToppIf access methods —

const char* const	**name** () const	*particle name*
float	**mass** () const	*mass*
float	**wid** () const	*width*
float	**trbw** () const	*truncate Breit-Wigner at mass - trbw*
float	**life** () const	*life time*
int	**chrg** () const	*3 * charge*
const ToppIf&	**anti** () const	*access to anti-particle*
int	**lund** () const	*LUND code*

— Access methods —

const ToppIf&

	topp () const	*topp table pointer*
bool	**valid** () const	*Do we represent a valid Topp entry?*
	— Factory functions —	
SP_sp	**swith** (const SProperty<TrackIf, bool>& rhs) const	*create a SPattern object*
PMC_sp	**createPattern** () const	
PMC_sp	**with** (const SProperty<TrackIf, bool>& rhs) const	
template <unsigned int NTR> typename DP_trait <NTR> ::DPMC_sp	**decaysTo** (const SPatternExpression<NTR>& rhs) const	*create a SDecayPatternMatch-Container object*
	— Operators —	
bool	**operator==** (const ToppIf& rhs) const	
bool	**operator==** (const GTopp& rhs) const	
bool	**operator!=** (const GTopp& rhs) const	
bool	**operator**< (const GTopp& rhs) const	
	— Expert functions —	
void	**set_topp** (const ArtePointer<Topp>& topp)	*set topp table pointer*
void	**set_anti** (const GTopp& anti)	*set link to anti GTopp object*
std::ostream&	**print** (std::ostream& os) const	*used by operator<<()*

Representation of a Topp table entry.

This class represents a Topp table entry via a pointer to the ArteTable<Topp>. In addition, factory functions are provided to generate objects which are needed for a pattern based analysis.

Author: T. Glebe

19

class **PID**

Definitions for Particle ID

Public Members

static const int
electron

static const int
muon

static const int
pion

static const int
kaon

static const int
proton

Definitions for Particle ID.

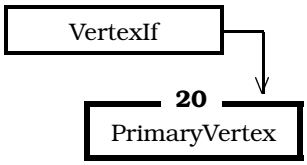
Author: T. Glebe

20

class **PrimaryVertex** : public VertexIf

Primary Vertex class

Inheritance



Public Members

— Constructors —

PrimaryVertex ()

virtual ~**PrimaryVertex** ()

— Typedefs —

typedef vector <TrackIf*>
t_vec

typedef t_vec::iterator
iterator

typedef t_vec::const_iterator
const_iterator

— VertexIf Access methods —

float	**vx** () const	v_x = *vertex* x *position*
float	**vy** () const	v_y = *vertex* y *position*
float	**vz** () const	v_z = *vertex* z *position*
float	**chi2** () const	χ^2 *of vertex fit*
float	**prob** () const	*upper tail* χ^2 *probability*
unsigned short int	**ndf** () const	*degrees of freedom of vertex fit*

unsigned short int

	ntracks () const	*no of tracks in vertex*
const TrackIf*		
	track (unsigned int i) const	*return a track pointer*
const TrackIf*&		
	track (unsigned int i)	*read/write track access*
SVector <double, 3>		
	vpos () const	*vertex position* $\vec{v} = (v_x,\ v_y,\ v_z)$
const SMatrix <double, 3> &		
	VCOV () const	*vertex covariance matrix*
float	**vtx_cov_x** () const	σ_x^2 *of vertex*
float	**vtx_cov_y** () const	σ_y^2 *of vertex*
float	**vtx_cov_z** () const	σ_z^2 *of vertex*
	— Expert methods —	
std::ostream&		
	print (std::ostream&) const	*used by operator<<()*
	— Expert functions —	
bool	**findVertexVt** ()	*do Kalman-Filter fit*
t_vec	**tracks**	*container for TrackIf pointers (public, for simplicity)*

Primary Vertex class.

This class is developed for the optimized fit of primary vertices. For primary vertices with $N_{\text{tracks}} \leq 20$ the high-performance vertexing class `SVertex` is used (see the `SMatrix` documentation). For vertices with more tracks the slow but dynamic-sized `vt++` vertexing class is used (see `Vt++` documentation).

Author: T. Glebe

21

class **AnalysisChain**

Class to manage execution of a series of AnalysisIf objects

Public Members

	— Constructors —	
	AnalysisChain ()	
virtual	~**AnalysisChain** ()	
	— Access methods —	
void	**push_back** (AnalysisIf& ana)	*add an analysis object*
bool	**EventSelect** () const	*dummy function: always return true*
bool	**EventCleanup** ()	*call EventCleanup() for each analysis job*
bool	**End** ()	*call End() for each analysis job*
bool	**DefineHistograms** ()	*call DefineHistograms() for each analysis job*
bool	**DefinePattern** ()	*call DefinePattern() for each analysis job*
bool	**Event** ()	*call Event() for each analysis job*
bool	**Event_Thread** ()	*start each Event() routine in separate thread*
bool	**ThreadEvent** ()	*start threads*

Class to manage execution of a series of AnalysisIf objects.

The AnalysisChain class can handle any analysis class which is derived from the `AnalysisIf` interface (see the `Interfaces` project).

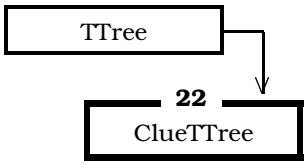
Author: T. Glebe

22

class **ClueTTree** : public TTree

Modified ROOT TTree class

Inheritance



Public Members

— Constructors —

ClueTTree ()

ClueTTree (const char *name,
const char *title,
Int_t maxvirtualsize=0)

virtual ~**ClueTTree** ()

— Access methods —

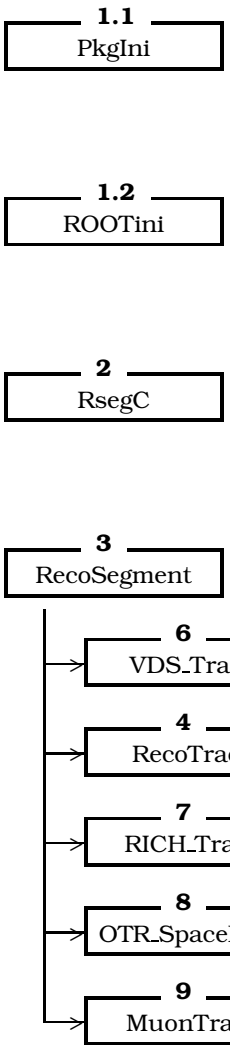
virtual TBranch*
Branch (const char *name,
const char *classname, void *addobj,
Int_t bufsize=32000, Int_t splitlevel=1)
fixed TTree function

Modified ROOT TTree class.

This TTree class fixes a bug in ROOT versions $\leq$ 2.25.03 which is essential for the STL compatible TClonesArrays of the `Gate` project. Since ROOT version 3.00 this bug is fixed in the official ROOT version.

Author: T. Glebe

Class Graph



PkgIni 5

ROOTini 6

RsegC 8

RecoSegment 11

VDS_Track 26

RecoTrack 18

RICH_Track 27

OTR_SpacePoint 28

MuonTrack 29

5 RecoVertex 22

10 TrackClones 30

11 EcalHit 32

12 MC_Track 38

13 MC_Vertex 44

14 StreamObject 47

15 RtEvent 48

16 Wire 53

17 Topp 55

18 GTopp 57

19 PID 59

20 PrimaryVertex 60

21 AnalysisChain 62

22 ClueTTree 63