

HepMC Reference Manual

2.05.01

Generated by Doxygen 1.4.7

Thu Jan 7 13:10:15 2010

Contents

1	HepMC Directory Hierarchy	1
1.1	HepMC Directories	1
2	HepMC Namespace Index	3
2.1	HepMC Namespace List	3
3	HepMC Hierarchical Index	5
3.1	HepMC Class Hierarchy	5
4	HepMC Class Index	7
4.1	HepMC Class List	7
5	HepMC File Index	9
5.1	HepMC File List	9
6	HepMC Directory Documentation	11
6.1	/home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/examples/ Directory Reference	11
6.2	/home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/fio/ Directory Reference	12
6.3	/home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/HepMC/ Directory Reference	13
6.4	/home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/src/ Directory Reference	14
6.5	/home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/test/ Directory Reference	15
7	HepMC Namespace Documentation	17
7.1	CLHEP Namespace Reference	17
7.2	detail Namespace Reference	18
7.3	HepMC Namespace Reference	19
7.4	HepMC::detail Namespace Reference	31
7.5	HepMC::Units Namespace Reference	35
7.6	Units Namespace Reference	37
8	HepMC Class Documentation	39

8.1	HepMC::detail::disable_if<, > Struct Template Reference	39
8.2	HepMC::detail::disable_if< false, T > Struct Template Reference	40
8.3	HepMC::detail::enable_if<, > Struct Template Reference	41
8.4	HepMC::detail::enable_if< true, T > Struct Template Reference	42
8.5	HepMC::Flow Class Reference	43
8.6	HepMC::FourVector Class Reference	50
8.7	HepMC::GenCrossSection Class Reference	60
8.8	HepMC::GenEvent Class Reference	64
8.9	HepMC::GenEvent::particle_const_iterator Class Reference	86
8.10	HepMC::GenEvent::particle_iterator Class Reference	89
8.11	HepMC::GenEvent::vertex_const_iterator Class Reference	92
8.12	HepMC::GenEvent::vertex_iterator Class Reference	95
8.13	HepMC::GenParticle Class Reference	99
8.14	HepMC::GenVertex Class Reference	109
8.15	HepMC::GenVertex::edge_iterator Class Reference	123
8.16	HepMC::GenVertex::particle_iterator Class Reference	126
8.17	HepMC::GenVertex::vertex_iterator Class Reference	129
8.18	HepMC::HeavyIon Class Reference	133
8.19	HepMC::HEPEVT_Wrapper Class Reference	141
8.20	HepMC::IO_AsciiParticles Class Reference	154
8.21	HepMC::IO_BaseClass Class Reference	158
8.22	HepMC::IO_Exception Class Reference	162
8.23	HepMC::IO_GenEvent Class Reference	164
8.24	HepMC::IO_HEPEVT Class Reference	169
8.25	HepMC::IO_HERWIG Class Reference	174
8.26	HepMC::IO_PDG_ParticleDataTable Class Reference	181
8.27	HepMC::detail::is_arithmetic< T > Struct Template Reference	184
8.28	HepMC::detail::is_arithmetic< char > Struct Template Reference	185
8.29	HepMC::detail::is_arithmetic< double > Struct Template Reference	186
8.30	HepMC::detail::is_arithmetic< float > Struct Template Reference	187
8.31	HepMC::detail::is_arithmetic< int > Struct Template Reference	188
8.32	HepMC::detail::is_arithmetic< long > Struct Template Reference	189
8.33	HepMC::detail::is_arithmetic< long double > Struct Template Reference	190
8.34	HepMC::detail::is_arithmetic< short > Struct Template Reference	191
8.35	HepMC::detail::is_arithmetic< signed char > Struct Template Reference	192
8.36	HepMC::detail::is_arithmetic< unsigned char > Struct Template Reference	193

8.37	HepMC::detail::is_arithmetic< unsigned int > Struct Template Reference	194
8.38	HepMC::detail::is_arithmetic< unsigned long > Struct Template Reference	195
8.39	HepMC::detail::is_arithmetic< unsigned short > Struct Template Reference	196
8.40	IsEventGood Class Reference	197
8.41	IsFinalState Class Reference	198
8.42	IsGoodEvent Class Reference	199
8.43	IsGoodEventMyPythia Class Reference	200
8.44	IsPhoton Class Reference	201
8.45	IsStateFinal Class Reference	202
8.46	IsW_Boson Class Reference	203
8.47	HepMC::ParticleData Class Reference	204
8.48	HepMC::ParticleDataTable Class Reference	211
8.49	HepMC::PdfInfo Class Reference	218
8.50	HepMC::Polarization Class Reference	225
8.51	HepMC::StreamInfo Class Reference	230
8.52	HepMC::TempParticleMap Class Reference	235
8.53	HepMC::ThreeVector Class Reference	238
8.54	HepMC::WeightContainer Class Reference	244
9	HepMC File Documentation	251
9.1	CompareGenEvent.cc File Reference	251
9.2	CompareGenEvent.h File Reference	252
9.3	enable_if.h File Reference	253
9.4	example_BuildEventFromScratch.cc File Reference	254
9.5	example_EventSelection.cc File Reference	255
9.6	example_MyHerwig.cc File Reference	256
9.7	example_MyPythia.cc File Reference	257
9.8	example_MyPythiaOnlyToHepMC.cc File Reference	260
9.9	example_PythiaStreamIO.cc File Reference	261
9.10	example_UsingIterators.cc File Reference	263
9.11	Flow.cc File Reference	264
9.12	Flow.h File Reference	265
9.13	GenCrossSection.cc File Reference	266
9.14	GenCrossSection.h File Reference	267
9.15	GenEvent.cc File Reference	268
9.16	GenEvent.h File Reference	269
9.17	GenEventStreamIO.cc File Reference	271

9.18	GenParticle.cc File Reference	273
9.19	GenParticle.h File Reference	274
9.20	GenVertex.cc File Reference	275
9.21	GenVertex.h File Reference	276
9.22	HeavyIon.cc File Reference	277
9.23	HeavyIon.h File Reference	278
9.24	HEPEVT_Wrapper.cc File Reference	279
9.25	HEPEVT_Wrapper.h File Reference	280
9.26	HepMCDefs.h File Reference	282
9.27	HerwigHelper.h File Reference	283
9.28	HerwigWrapper.h File Reference	284
9.29	HerwigWrapper6_4.h File Reference	285
9.30	initPythia.cc File Reference	302
9.31	IO_AsciiParticles.cc File Reference	303
9.32	IO_AsciiParticles.h File Reference	304
9.33	IO_BaseClass.h File Reference	305
9.34	IO_Exception.h File Reference	306
9.35	IO_GenEvent.cc File Reference	307
9.36	IO_GenEvent.h File Reference	308
9.37	IO_HEPEVT.cc File Reference	309
9.38	IO_HEPEVT.h File Reference	310
9.39	IO_HERWIG.cc File Reference	311
9.40	IO_HERWIG.h File Reference	312
9.41	IO_PDG_ParticleDataTable.cc File Reference	313
9.42	IO_PDG_ParticleDataTable.h File Reference	314
9.43	is_arithmetic.h File Reference	315
9.44	IsGoodEvent.h File Reference	316
9.45	list_of_examples.cc File Reference	317
9.46	list_of_examples.cc File Reference	318
9.47	ParticleData.cc File Reference	319
9.48	ParticleData.h File Reference	320
9.49	ParticleDataTable.h File Reference	321
9.50	PdfInfo.cc File Reference	322
9.51	PdfInfo.h File Reference	323
9.52	Polarization.cc File Reference	324
9.53	Polarization.h File Reference	325

9.54	PythiaHelper.h File Reference	326
9.55	PythiaWrapper.h File Reference	327
9.56	PythiaWrapper5_720.h File Reference	328
9.57	PythiaWrapper6_152.h File Reference	336
9.58	PythiaWrapper6_152_WIN32.h File Reference	343
9.59	PythiaWrapper6_2.h File Reference	344
9.60	PythiaWrapper6_2_WIN32.h File Reference	352
9.61	SearchVector.cc File Reference	353
9.62	SearchVector.h File Reference	354
9.63	SimpleVector.h File Reference	355
9.64	StreamHelpers.cc File Reference	356
9.65	StreamHelpers.h File Reference	357
9.66	StreamInfo.cc File Reference	358
9.67	StreamInfo.h File Reference	359
9.68	TempParticleMap.h File Reference	360
9.69	testFlow.cc File Reference	361
9.70	testHepMCIteration.h File Reference	362
9.71	testHepMCMethods.cc File Reference	363
9.72	testHepMCMethods.h File Reference	364
9.73	testHerwigCopies.cc File Reference	365
9.74	testPrintBug.cc File Reference	366
9.75	testPythiaCopies.cc File Reference	367
9.76	testSimpleVector.cc File Reference	368
9.77	testUnits.cc File Reference	369
9.78	Units.h File Reference	370
9.79	VectorConversion.h File Reference	371
9.80	Version.h File Reference	372
9.81	WeightContainer.h File Reference	373
10	HepMC Example Documentation	375
10.1	example_BuildEventFromScratch.cc	375
10.2	example_EventSelection.cc	378
10.3	example_MyHerwig.cc	380
10.4	example_MyPythia.cc	382
10.5	example_MyPythiaOnlyToHepMC.cc	387
10.6	example_PythiaStreamIO.cc	389
10.7	example_UsingIterators.cc	392

10.8 testFlow.cc	395
10.9 testHepMC.cc.in	398
10.10testHepMCIteration.cc.in	403
10.11testHerwigCopies.cc	406
10.12testMass.cc.in	408
10.13testMultipleCopies.cc.in	411
10.14testPrintBug.cc	414
10.15testPythiaCopies.cc	415
10.16testSimpleVector.cc	417
10.17testStreamIO.cc.in	420
10.18testUnits.cc	425
10.19VectorConversion.h	427

Chapter 1

HepMC Directory Hierarchy

1.1 HepMC Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

examples	11
fio	12
HepMC	13
src	14
test	15

Chapter 2

HepMC Namespace Index

2.1 HepMC Namespace List

Here is a list of all namespaces with brief descriptions:

CLHEP	17
detail	18
HepMC	19
HepMC::detail	31
HepMC::Units	35
Units	37

Chapter 3

HepMC Hierarchical Index

3.1 HepMC Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

HepMC::detail::disable_if<, >	39
HepMC::detail::disable_if< false, T >	40
HepMC::detail::enable_if<, >	41
HepMC::detail::enable_if< true, T >	42
std::exception	
std::runtime_error	
HepMC::IO_Exception	162
HepMC::Flow	43
HepMC::FourVector	50
HepMC::GenCrossSection	60
HepMC::GenEvent	64
HepMC::GenEvent::particle_const_iterator	86
HepMC::GenEvent::particle_iterator	89
HepMC::GenEvent::vertex_const_iterator	92
HepMC::GenEvent::vertex_iterator	95
HepMC::GenParticle	99
HepMC::GenVertex	109
HepMC::GenVertex::edge_iterator	123
HepMC::GenVertex::particle_iterator	126
HepMC::GenVertex::vertex_iterator	129
HepMC::HeavyIon	133
HepMC::HEPEVT_Wrapper	141
HepMC::IO_BaseClass	158
HepMC::IO_AsciiParticles	154
HepMC::IO_GenEvent	164
HepMC::IO_HEPEVT	169
HepMC::IO_HERWIG	174
HepMC::IO_PDG_ParticleDataTable	181
HepMC::detail::is_arithmetic< T >	184
HepMC::detail::is_arithmetic< char >	185
HepMC::detail::is_arithmetic< double >	186
HepMC::detail::is_arithmetic< float >	187
HepMC::detail::is_arithmetic< int >	188

HepMC::detail::is_arithmetic< long >	189
HepMC::detail::is_arithmetic< long double >	190
HepMC::detail::is_arithmetic< short >	191
HepMC::detail::is_arithmetic< signed char >	192
HepMC::detail::is_arithmetic< unsigned char >	193
HepMC::detail::is_arithmetic< unsigned int >	194
HepMC::detail::is_arithmetic< unsigned long >	195
HepMC::detail::is_arithmetic< unsigned short >	196
IsEventGood	197
IsFinalState	198
IsGoodEvent	199
IsGoodEventMyPythia	200
IsPhoton	201
IsStateFinal	202
IsW_Boson	203
HepMC::ParticleData	204
HepMC::ParticleDataTable	211
HepMC::PdfInfo	218
HepMC::Polarization	225
HepMC::StreamInfo	230
HepMC::TempParticleMap	235
HepMC::ThreeVector	238
HepMC::WeightContainer	244

Chapter 4

HepMC Class Index

4.1 HepMC Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

HepMC::detail::disable_if<, > (Internal - used by SimpleVector to decide if a class is arithmetic)	39
HepMC::detail::disable_if< false, T > (Internal - used by SimpleVector to decide if a class is arithmetic)	40
HepMC::detail::enable_if<, > (Internal - used to decide if a class is arithmetic)	41
HepMC::detail::enable_if< true, T > (Internal - use if class T is arithmetic)	42
HepMC::Flow (The flow object)	43
HepMC::FourVector (FourVector (p. 50) is a simple representation of a physics 4 vector)	50
HepMC::GenCrossSection (The GenCrossSection (p. 60) class stores the generated cross section)	60
HepMC::GenEvent (The GenEvent (p. 64) class is the core of HepMC (p. 19))	64
HepMC::GenEvent::particle_const_iterator (Const particle iterator)	86
HepMC::GenEvent::particle_iterator (Non-const particle iterator)	89
HepMC::GenEvent::vertex_const_iterator (Const vertex iterator)	92
HepMC::GenEvent::vertex_iterator (Non-const vertex iterator)	95
HepMC::GenParticle (The GenParticle (p. 99) class contains information about generated particles)	99
HepMC::GenVertex (GenVertex (p. 109) contains information about decay vertices)	109
HepMC::GenVertex::edge_iterator (Edge iterator)	123
HepMC::GenVertex::particle_iterator (Particle iterator)	126
HepMC::GenVertex::vertex_iterator (Vertex iterator)	129
HepMC::HeavyIon (The HeavyIon (p. 133) class stores information about heavy ions)	133
HepMC::HEPEVT_Wrapper (Generic Wrapper for the fortran HEPEVT common block)	141
HepMC::IO_AsciiParticles (Event input/output in ascii format for eye and machine reading)	154
HepMC::IO_BaseClass (All input/output classes inherit from IO_BaseClass (p. 158))	158
HepMC::IO_Exception (IO exception handling)	162
HepMC::IO_GenEvent (IO_GenEvent (p. 164) also deals with HeavyIon (p. 133) and Pdf-Info (p. 218))	164
HepMC::IO_HEPEVT (HEPEVT IO class)	169
HepMC::IO_HERWIG (IO_HERWIG (p. 174) is used to get Herwig information)	174
HepMC::IO_PDG_ParticleDataTable (Example ParticleDataTable (p. 211) IO method)	181
HepMC::detail::is_arithmetic< T > (Undefined and therefore non-arithmetic)	184
HepMC::detail::is_arithmetic< char > (Character is arithmetic)	185

HepMC::detail::is_arithmetic< double > (Double is arithmetic)	186
HepMC::detail::is_arithmetic< float > (Float is arithmetic)	187
HepMC::detail::is_arithmetic< int > (Int is arithmetic)	188
HepMC::detail::is_arithmetic< long > (Long is arithmetic)	189
HepMC::detail::is_arithmetic< long double > (Long double is arithmetic)	190
HepMC::detail::is_arithmetic< short > (Short is arithmetic)	191
HepMC::detail::is_arithmetic< signed char > (Signed character is arithmetic)	192
HepMC::detail::is_arithmetic< unsigned char > (Unsigned character is arithmetic)	193
HepMC::detail::is_arithmetic< unsigned int > (Unsigned int is arithmetic)	194
HepMC::detail::is_arithmetic< unsigned long > (Unsigned long is arithmetic)	195
HepMC::detail::is_arithmetic< unsigned short > (Unsigned short is arithmetic)	196
IsEventGood (Example class)	197
IsFinalState (Test class)	198
IsGoodEvent (Used in the tests)	199
IsGoodEventMyPythia (Example class)	200
IsPhoton (Example class)	201
IsStateFinal (Example class)	202
IsW_Boson (Example class)	203
HepMC::ParticleData (Example ParticleData (p. 204) class)	204
HepMC::ParticleDataTable (Example ParticleDataTable (p. 211) class)	211
HepMC::PdfInfo (The PdfInfo (p. 218) class stores PDF information)	218
HepMC::Polarization (The Polarization (p. 225) class stores theta and phi for a GenParticle (p. 99))	225
HepMC::StreamInfo (StreamInfo (p. 230) contains extra information needed when using streaming IO)	230
HepMC::TempParticleMap (TempParticleMap (p. 235) is a temporary GenParticle* container used during input)	235
HepMC::ThreeVector (ThreeVector (p. 238) is a simple representation of a position or displacement 3 vector)	238
HepMC::WeightContainer (Container for the Weights associated with an event or vertex)	244

Chapter 5

HepMC File Index

5.1 HepMC File List

Here is a list of all files with brief descriptions:

CompareGenEvent.cc	251
CompareGenEvent.h	252
enable_if.h	253
example_BuildEventFromScratch.cc	254
example_EventSelection.cc	255
example_MyHerwig.cc	256
example_MyPythia.cc	257
example_MyPythiaOnlyToHepMC.cc	260
example_PythiaStreamIO.cc	261
example_UsingIterators.cc	263
Flow.cc	264
Flow.h	265
GenCrossSection.cc	266
GenCrossSection.h	267
GenEvent.cc	268
GenEvent.h	269
GenEventStreamIO.cc	271
GenParticle.cc	273
GenParticle.h	274
GenVertex.cc	275
GenVertex.h	276
HeavyIon.cc	277
HeavyIon.h	278
HEPEVT_Wrapper.cc	279
HEPEVT_Wrapper.h	280
HepMCDefs.h	282
HerwigHelper.h	283
HerwigWrapper.h	284
HerwigWrapper6_4.h	285
initPythia.cc	302
IO_AsciiParticles.cc	303
IO_AsciiParticles.h	304
IO_BaseClass.h	305

IO_Exception.h	306
IO_GenEvent.cc	307
IO_GenEvent.h	308
IO_HEPEVT.cc	309
IO_HEPEVT.h	310
IO_HERWIG.cc	311
IO_HERWIG.h	312
IO_PDG_ParticleDataTable.cc	313
IO_PDG_ParticleDataTable.h	314
is_arithmetic.h	315
IsGoodEvent.h	316
examples/list_of_examples.cc	317
test/list_of_examples.cc	318
ParticleData.cc	319
ParticleData.h	320
ParticleDataTable.h	321
PdfInfo.cc	322
PdfInfo.h	323
Polarization.cc	324
Polarization.h	325
PythiaHelper.h	326
PythiaWrapper.h	327
PythiaWrapper5_720.h	328
PythiaWrapper6_152.h	336
PythiaWrapper6_152_WIN32.h	343
PythiaWrapper6_2.h	344
PythiaWrapper6_2_WIN32.h	352
SearchVector.cc	353
SearchVector.h	354
SimpleVector.h	355
StreamHelpers.cc	356
StreamHelpers.h	357
StreamInfo.cc	358
StreamInfo.h	359
TempParticleMap.h	360
testFlow.cc	361
testHepMCIteration.h	362
testHepMCMethods.cc	363
testHepMCMethods.h	364
testHerwigCopies.cc	365
testPrintBug.cc	366
testPythiaCopies.cc	367
testSimpleVector.cc	368
testUnits.cc	369
Units.h	370
VectorConversion.h	371
Version.h	372
WeightContainer.h	373

Chapter 6

HepMC Directory Documentation

6.1 /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/examples/ Directory Reference

Files

- file `example_BuildEventFromScratch.cc`
- file `example_EventSelection.cc`
- file `example_MyHerwig.cc`
- file `example_MyPythia.cc`
- file `example_MyPythiaOnlyToHepMC.cc`
- file `example_PythiaStreamIO.cc`
- file `example_UsingIterators.cc`
- file `HerwigHelper.h`
- file `initPythia.cc`
- file `examples/list_of_examples.cc`
- file `PythiaHelper.h`
- file `testHerwigCopies.cc`
- file `testPythiaCopies.cc`
- file `VectorConversion.h`

6.2 /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/fio/ Directory Reference

Files

- file **HEPEVT_Wrapper.cc**
- file **IO_HEPEVT.cc**
- file **IO_HERWIG.cc**

6.3 /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/HepMC/ Directory Reference

Files

- file CompareGenEvent.h
- file enable_if.h
- file Flow.h
- file GenCrossSection.h
- file GenEvent.h
- file GenParticle.h
- file GenVertex.h
- file HeavyIon.h
- file HEPEVT_Wrapper.h
- file HepMCDefs.h
- file HerwigWrapper.h
- file HerwigWrapper6_4.h
- file IO_AsciiParticles.h
- file IO_BaseClass.h
- file IO_Exception.h
- file IO_GenEvent.h
- file IO_HEPEVT.h
- file IO_HERWIG.h
- file IO_PDG_ParticleDataTable.h
- file is_arithmetic.h
- file ParticleData.h
- file ParticleDataTable.h
- file PdfInfo.h
- file Polarization.h
- file PythiaWrapper.h
- file PythiaWrapper5_720.h
- file PythiaWrapper6_152.h
- file PythiaWrapper6_152_WIN32.h
- file PythiaWrapper6_2.h
- file PythiaWrapper6_2_WIN32.h
- file SearchVector.h
- file SimpleVector.h
- file StreamHelpers.h
- file StreamInfo.h
- file TempParticleMap.h
- file Units.h
- file Version.h
- file WeightContainer.h

6.4 /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/src/ Directory Reference

Files

- file CompareGenEvent.cc
- file Flow.cc
- file GenCrossSection.cc
- file GenEvent.cc
- file GenEventStreamIO.cc
- file GenParticle.cc
- file GenVertex.cc
- file HeavyIon.cc
- file IO_AsciiParticles.cc
- file IO_GenEvent.cc
- file IO_PDG_ParticleDataTable.cc
- file ParticleData.cc
- file PdfInfo.cc
- file Polarization.cc
- file SearchVector.cc
- file StreamHelpers.cc
- file StreamInfo.cc

6.5 /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/test/ Directory Reference

Files

- file `IsGoodEvent.h`
- file `test/list_of_examples.cc`
- file `testFlow.cc`
- file `testHepMCIteration.h`
- file `testHepMCMethods.cc`
- file `testHepMCMethods.h`
- file `testPrintBug.cc`
- file `testSimpleVector.cc`
- file `testUnits.cc`

Chapter 7

HepMC Namespace Documentation

7.1 CLHEP Namespace Reference

7.1.1 Detailed Description

CLHEP (p. 17) Vector classes are used in one of the examples

7.2 detail Namespace Reference

7.2.1 Detailed Description

internal namespace

7.3 HepMC Namespace Reference

Classes

- class **Flow**
The flow object.
- class **GenCrossSection**
The GenCrossSection (p. 60) class stores the generated cross section.
- class **GenEvent**
The GenEvent (p. 64) class is the core of HepMC (p. 19).
- class **GenParticle**
The GenParticle (p. 99) class contains information about generated particles.
- class **GenVertex**
GenVertex (p. 109) contains information about decay vertices.
- class **HeavyIon**
The HeavyIon (p. 133) class stores information about heavy ions.
- class **HEPEVT_Wrapper**
Generic Wrapper for the fortran HEPEVT common block.
- class **IO_AsciiParticles**
event input/output in ascii format for eye and machine reading
- class **IO_BaseClass**
all input/output classes inherit from IO_BaseClass (p. 158)
- class **IO_Exception**
IO exception handling.
- class **IO_GenEvent**
IO_GenEvent (p. 164) also deals with HeavyIon (p. 133) and PdfInfo (p. 218).
- class **IO_HEPEVT**
HEPEVT IO class.
- class **IO_HERWIG**
IO_HERWIG (p. 174) is used to get Herwig information.
- class **IO_PDG_ParticleDataTable**
an example ParticleDataTable (p. 211) IO method
- class **ParticleData**
an example ParticleData (p. 204) class
- class **ParticleDataTable**

an example ParticleDataTable (p. 211) class

- **class PdfInfo**

The PdfInfo (p. 218) class stores PDF information.

- **class Polarization**

The Polarization (p. 225) class stores theta and phi for a GenParticle (p. 99).

- **class FourVector**

FourVector (p. 50) is a simple representation of a physics 4 vector.

- **class ThreeVector**

ThreeVector (p. 238) is a simple representation of a position or displacement 3 vector.

- **class StreamInfo**

StreamInfo (p. 230) contains extra information needed when using streaming IO.

- **class TempParticleMap**

TempParticleMap (p. 235) is a temporary GenParticle container used during input.*

- **class WeightContainer**

Container for the Weights associated with an event or vertex.

Namespaces

- namespace **detail**
- namespace **Units**

Enumerations

- enum **IteratorRange** {
parents, children, family, ancestors,
descendants, relatives }
type of iteration

- enum **known_io** {
gen = 1, ascii, extascii, ascii_pdt,
extascii_pdt }

The known_io enum is used to track which type of input is being read.

Functions

- **bool compareGenEvent (GenEvent *, GenEvent *)**
- **bool compareSignalProcessVertex (GenEvent *, GenEvent *)**
- **bool compareBeamParticles (GenEvent *, GenEvent *)**
- **bool compareWeights (GenEvent *, GenEvent *)**

- **bool compareVertices (GenEvent *, GenEvent *)**
- **bool compareParticles (GenEvent *, GenEvent *)**
- **bool compareVertex (GenVertex *v1, GenVertex *v2)**
- **std::ostream & operator<< (std::ostream &os, GenCrossSection &xs)**
- **std::istream & operator>> (std::istream &is, GenCrossSection &xs)**
- **template<class InputIterator, class OutputIterator, class Predicate> void copy_if (InputIterator first, InputIterator last, OutputIterator out, Predicate pred)**
define the type of iterator to use
- **std::ostream & operator<< (std::ostream &, GenEvent &)**
standard streaming IO output operator
- **std::istream & operator>> (std::istream &, GenEvent &)**
standard streaming IO input operator
- **std::istream & set_input_units (std::istream &, Units::MomentumUnit, Units::LengthUnit)**
set the units for this input stream
- **std::ostream & write_HepMC_IO_block_begin (std::ostream &)**
Explicitly write the begin block lines that IO_GenEvent (p. 164) uses.
- **std::ostream & write_HepMC_IO_block_end (std::ostream &)**
Explicitly write the end block line that IO_GenEvent (p. 164) uses.
- **GenEvent & convert_units (GenEvent &evt, Units::MomentumUnit m, Units::LengthUnit l)**
- **std::ostream & operator<< (std::ostream &, HeavyIon const *)**
Write the contents of HeavyIon (p. 133) to an output stream.
- **std::istream & operator>> (std::istream &, HeavyIon *)**
Read the contents of HeavyIon (p. 133) from an input stream.
- **double clifetime_from_width (double width)**
set lifetime from width
- **std::ostream & operator<< (std::ostream &, PdfInfo const *)**
- **std::istream & operator>> (std::istream &, PdfInfo *)**
- **bool not_in_vector (std::vector< HepMC::GenParticle * > *, GenParticle *)**
returns true if it cannot find GenParticle in the vector*
- **std::vector< HepMC::GenParticle * >::iterator already_in_vector (std::vector< GenParticle * > *v, GenParticle *p)**
returns true if GenParticle (p. 99) is in the vector
- **void version ()**
print HepMC (p. 19) version
- **void writeVersion (std::ostream &os)**
write HepMC (p. 19) version to os
- **std::string versionName ()**

return HepMC (p. 19) version

- **std::ostream & operator<< (std::ostream &ostr, const Flow &f)**
for printing
- **void HepMCStreamCallback (std::ios_base::event e, std::ios_base &b, int i)**
- **template<class IO> StreamInfo & get_stream_info (IO &iost)**
- **std::ostream & establish_output_stream_info (std::ostream &os)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & establish_input_stream_info (std::istream &is)**
used by IO_GenEvent (p. 164) constructor
- **std::ostream & operator<< (std::ostream &ostr, const GenParticle &part)**
print particle
- **std::ostream & operator<< (std::ostream &ostr, const GenVertex &vtx)**
print vertex information
- **std::ostream & operator<< (std::ostream &ostr, const ParticleData &pdata)**
write to ostr
- **std::ostream & operator<< (std::ostream &ostr, const Polarization &polar)**
print polarization information

Variables

- static const double **HepMC_hbarc**
 *$\hbar c \rightarrow$ calculated with units of [mm*GeV]*
- static const double **HepMC_pi** = 3.14159265358979323846

7.3.1 Detailed Description

All classes in the **HepMC** (p. 19) packages are in the **HepMC** (p. 19) namespace

7.3.2 Enumeration Type Documentation

7.3.2.1 enum HepMC::IteratorRange

type of iteration

Enumerator:

parents
children
family
ancestors

descendants

relatives

Definition at line 35 of file GenVertex.h.

7.3.2.2 enum HepMC::known_io

The known_io enum is used to track which type of input is being read.

Enumerator:

gen

ascii

extascii

ascii_pdt

extascii_pdt

Definition at line 17 of file StreamInfo.h.

7.3.3 Function Documentation

7.3.3.1 bool HepMC::compareGenEvent (GenEvent *, GenEvent *)

Examples:

`testHerwigCopies.cc`, `testMultipleCopies.cc.in`, and `testPythiaCopies.cc`.

Definition at line 16 of file CompareGenEvent.cc.

References `HepMC::GenEvent::alphaQCD()`, `HepMC::GenEvent::alphaQED()`, `compareBeamParticles()`, `compareParticles()`, `compareSignalProcessVertex()`, `compareVertices()`, `compareWeights()`, `HepMC::GenEvent::event_number()`, `HepMC::GenEvent::event_scale()`, `HepMC::GenEvent::heavy_ion()`, `HepMC::GenEvent::mpi()`, `HepMC::GenEvent::pdf_info()`, `HepMC::GenEvent::random_states()`, and `HepMC::GenEvent::signal_process_id()`.

Referenced by `main()`.

7.3.3.2 bool HepMC::compareSignalProcessVertex (GenEvent *, GenEvent *)

Definition at line 64 of file CompareGenEvent.cc.

References `HepMC::GenEvent::signal_process_vertex()`.

Referenced by `compareGenEvent()`.

7.3.3.3 bool HepMC::compareBeamParticles (GenEvent *, GenEvent *)

Definition at line 77 of file CompareGenEvent.cc.

References `HepMC::GenEvent::beam_particles()`.

Referenced by `compareGenEvent()`.

7.3.3.4 bool HepMC::compareWeights (GenEvent *, GenEvent *)

Definition at line 92 of file CompareGenEvent.cc.

References HepMC::WeightContainer::size(), and HepMC::GenEvent::weights().

Referenced by compareGenEvent().

7.3.3.5 bool HepMC::compareVertices (GenEvent *, GenEvent *)

Definition at line 131 of file CompareGenEvent.cc.

References HepMC::GenEvent::barcode_to_vertex(), compareVertex(), v, HepMC::GenEvent::vertices_begin(), HepMC::GenEvent::vertices_end(), and HepMC::GenEvent::vertices_size().

Referenced by compareGenEvent().

7.3.3.6 bool HepMC::compareParticles (GenEvent *, GenEvent *)

Definition at line 109 of file CompareGenEvent.cc.

References HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), and HepMC::GenEvent::particles_size().

Referenced by compareGenEvent().

7.3.3.7 bool HepMC::compareVertex (GenVertex * v1, GenVertex * v2)

Definition at line 152 of file CompareGenEvent.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::particles_in_const_begin(), HepMC::GenVertex::particles_in_const_end(), HepMC::GenVertex::particles_in_size(), HepMC::GenVertex::particles_out_const_begin(), HepMC::GenVertex::particles_out_const_end(), HepMC::GenVertex::particles_out_size(), and HepMC::GenVertex::position().

Referenced by compareVertices().

7.3.3.8 std::ostream& HepMC::operator<< (std::ostream & os, GenCrossSection & xs)
[inline]

Definition at line 89 of file GenCrossSection.h.

References HepMC::GenCrossSection::write().

7.3.3.9 std::istream& HepMC::operator>> (std::istream & is, GenCrossSection & xs)
[inline]

Definition at line 92 of file GenCrossSection.h.

References HepMC::GenCrossSection::read().

7.3.3.10 `template<class InputIterator, class OutputIterator, class Predicate> void
HepMC::copy_if (InputIterator first, InputIterator last, OutputIterator out, Predicate
pred)`

define the type of iterator to use

Examples:

example_UsingIterators.cc, and testHepMCIteration.cc.in.

Definition at line 50 of file GenEvent.h.

Referenced by `main()`.

7.3.3.11 `std::ostream & HepMC::operator<< (std::ostream &, GenEvent &)`

standard streaming IO output operator

Writes evt to an output stream.

Definition at line 342 of file GenEventStreamIO.cc.

References `HepMC::GenEvent::write()`.

7.3.3.12 `std::istream & HepMC::operator>> (std::istream &, GenEvent &)`

standard streaming IO input operator

Definition at line 349 of file GenEventStreamIO.cc.

References `HepMC::GenEvent::read()`.

7.3.3.13 `std::istream & HepMC::set_input_units (std::istream &, Units::MomentumUnit,
Units::LengthUnit)`

set the units for this input stream

Examples:

testStreamIO.cc.in.

Definition at line 357 of file GenEventStreamIO.cc.

References `get_stream_info()`, and `HepMC::StreamInfo::use_input_units()`.

Referenced by `HepMC::IO_GenEvent::use_input_units()`.

7.3.3.14 `std::ostream & HepMC::write_HepMC_IO_block_begin (std::ostream &)`

Explicitly write the begin block lines that `IO_GenEvent` (p. 164) uses.

Examples:

example_PythiaStreamIO.cc, and testStreamIO.cc.in.

Definition at line 369 of file GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), get_stream_info(), HepMC::StreamInfo::IO_GenEvent_Key(), and versionName().

Referenced by readPythiaStreamIO(), HepMC::IO_GenEvent::write_event(), and writePythiaStreamIO().

7.3.3.15 std::ostream & HepMC::write_HepMC_IO_block_end (std::ostream &)

Explicitly write the end block line that **IO_GenEvent** (p. 164) uses.

Examples:

example_PythiaStreamIO.cc, and testStreamIO.cc.in.

Definition at line 382 of file GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), get_stream_info(), and HepMC::StreamInfo::IO_GenEvent_End().

Referenced by readPythiaStreamIO(), HepMC::IO_GenEvent::write_comment(), writePythiaStreamIO(), and HepMC::IO_GenEvent::~IO_GenEvent().

7.3.3.16 GenEvent& HepMC::convert_units (GenEvent & evt, Units::MomentumUnit m, Units::LengthUnit l) [inline]

Definition at line 637 of file GenEvent.h.

References HepMC::GenEvent::use_units().

7.3.3.17 std::ostream & HepMC::operator<< (std::ostream & os, HeavyIon const * ion)

Write the contents of **HeavyIon** (p. 133) to an output stream.

Write the contents of **HeavyIon** (p. 133) to an output stream. **GenEvent** (p. 64) stores a pointer to a **HeavyIon** (p. 133).

Definition at line 23 of file HeavyIon.cc.

References HepMC::HeavyIon::eccentricity(), HepMC::HeavyIon::event_plane_angle(), HepMC::HeavyIon::impact_parameter(), HepMC::HeavyIon::N_Nwounded_collisions(), HepMC::HeavyIon::Ncoll(), HepMC::HeavyIon::Ncoll_hard(), HepMC::HeavyIon::Npart_proj(), HepMC::HeavyIon::Npart_targ(), HepMC::HeavyIon::Nwounded_N_collisions(), HepMC::HeavyIon::Nwounded_Nwounded_collisions(), HepMC::detail::output(), HepMC::HeavyIon::sigma_inel_NN(), HepMC::HeavyIon::spectator_neutrons(), and HepMC::HeavyIon::spectator_protons().

7.3.3.18 std::istream & HepMC::operator>> (std::istream & is, HeavyIon * ion)

Read the contents of **HeavyIon** (p. 133) from an input stream.

Read the contents of **HeavyIon** (p. 133) from an input stream. **GenEvent** (p. 64) stores a pointer to a **HeavyIon** (p. 133).

Definition at line 71 of file HeavyIon.cc.

References HepMC::HeavyIon::set_eccentricity(), HepMC::HeavyIon::set_event_plane_angle(), HepMC::HeavyIon::set_impact_parameter(), HepMC::HeavyIon::set_N_Nwounded_collisions(), HepMC::HeavyIon::set_Ncoll(), HepMC::HeavyIon::set_Ncoll_hard(), HepMC::HeavyIon::set_Npart_proj(), HepMC::HeavyIon::set_Npart_targ(), HepMC::HeavyIon::set_Nwounded_N_collisions(), HepMC::HeavyIon::set_Nwounded_Nwounded_collisions(), HepMC::HeavyIon::set_sigma_inel_NN(), HepMC::HeavyIon::set_spectator_neutrons(), and HepMC::HeavyIon::set_spectator_protons().

7.3.3.19 double HepMC::clifetime_from_width (double *width*)

set lifetime from width

if you want to instantiate the particle lifetime from its width, use this static method inside the constructor:

Examples:

example_BuildEventFromScratch.cc.

Definition at line 110 of file ParticleData.cc.

References HepMC_hbarc.

Referenced by main(), and HepMC::IO_PDG_ParticleDataTable::read_entry().

7.3.3.20 std::ostream & HepMC::operator<< (std::ostream &, PdfInfo const *)

Definition at line 21 of file PdfInfo.cc.

References HepMC::PdfInfo::id1(), HepMC::PdfInfo::id2(), HepMC::detail::output(), HepMC::PdfInfo::pdf1(), HepMC::PdfInfo::pdf2(), HepMC::PdfInfo::pdf_id1(), HepMC::PdfInfo::pdf_id2(), HepMC::PdfInfo::scalePDF(), HepMC::PdfInfo::x1(), and HepMC::PdfInfo::x2().

7.3.3.21 std::istream & HepMC::operator>> (std::istream &, PdfInfo *)

Definition at line 59 of file PdfInfo.cc.

References HepMC::PdfInfo::set_id1(), HepMC::PdfInfo::set_id2(), HepMC::PdfInfo::set_pdf1(), HepMC::PdfInfo::set_pdf2(), HepMC::PdfInfo::set_pdf_id1(), HepMC::PdfInfo::set_pdf_id2(), HepMC::PdfInfo::set_scalePDF(), HepMC::PdfInfo::set_x1(), and HepMC::PdfInfo::set_x2().

7.3.3.22 bool HepMC::not_in_vector (std::vector< HepMC::GenParticle * > *, GenParticle *)

returns true if it cannot find GenParticle* in the vector

Definition at line 11 of file SearchVector.cc.

References already_in_vector(), and p.

Referenced by HepMC::Flow::connected_partners(), and HepMC::Flow::dangling_connected_partners().

7.3.3.23 std::vector< HepMC::GenParticle * >::iterator HepMC::already_in_vector (std::vector< HepMC::GenParticle * > *, GenParticle *)

returns true if **GenParticle** (p.99) is in the vector

Returns the index of a `GenParticle*` within a vector. Returns -1 if `GenParticle*` is not in the vector.

Definition at line 18 of file `SearchVector.cc`.

References `p`.

Referenced by `not_in_vector()`, `HepMC::GenVertex::remove_particle_in()`, and `HepMC::GenVertex::remove_particle_out()`.

7.3.3.24 `void HepMC::version ()` [inline]

print **HepMC** (p. 19) version

Examples:

`testMass.cc.in`.

Definition at line 27 of file `Version.h`.

References `versionName()`.

7.3.3.25 `void HepMC::writeVersion (std::ostream & os)` [inline]

write **HepMC** (p. 19) version to `os`

Definition at line 33 of file `Version.h`.

References `versionName()`.

Referenced by `HepMC::GenEvent::print_version()`.

7.3.3.26 `std::string HepMC::versionName ()` [inline]

return **HepMC** (p. 19) version

Definition at line 22 of file `Version.h`.

References `HEPMC_VERSION`.

Referenced by `version()`, `HepMC::IO_AsciiParticles::write_event()`, `write_HepMC_IO_block_begin()`, and `writeVersion()`.

7.3.3.27 `std::ostream& HepMC::operator<< (std::ostream & ostr, const Flow & f)`

for printing

Definition at line 190 of file `Flow.cc`.

References `HepMC::Flow::m_icode`.

7.3.3.28 `void HepMC::HepMCStreamCallback (std::ios_base::event e, std::ios_base & b, int i)`

This method is called by the stream destructor. It does cleanup on stored user data (**StreamInfo** (p. 230)) and is registered by the first call to `get_stream_info()` (p. 29).

Definition at line 29 of file `GenEventStreamIO.cc`.

References `HepMC::StreamInfo::stream_id()`.

Referenced by `get_stream_info()`.

7.3.3.29 `template<class IO> StreamInfo& HepMC::get_stream_info (IO & iost)`

A custom iomanip that allows us to store and access user data (**StreamInfo** (p. 230)) associated with the stream. This method creates the **StreamInfo** (p. 230) object the first time it is called.

Definition at line 51 of file `GenEventStreamIO.cc`.

References `HepMCStreamCallback()`.

Referenced by `HepMC::detail::establish_input_stream_info()`, `establish_input_stream_info()`, `HepMC::detail::establish_output_stream_info()`, `establish_output_stream_info()`, `HepMC::GenEvent::read()`, `HepMC::detail::read_particle()`, `HepMC::detail::read_units()`, `set_input_units()`, `HepMC::GenEvent::write()`, `write_HepMC_IO_block_begin()`, and `write_HepMC_IO_block_end()`.

7.3.3.30 `std::ostream& HepMC::establish_output_stream_info (std::ostream & os)`

used by `IO_GenEvent` (p. 164) constructor

Definition at line 506 of file `GenEventStreamIO.cc`.

References `HepMC::StreamInfo::finished_first_event()`, and `get_stream_info()`.

Referenced by `HepMC::IO_GenEvent::IO_GenEvent()`.

7.3.3.31 `std::istream& HepMC::establish_input_stream_info (std::istream & is)`

used by `IO_GenEvent` (p. 164) constructor

Definition at line 520 of file `GenEventStreamIO.cc`.

References `HepMC::StreamInfo::finished_first_event()`, and `get_stream_info()`.

Referenced by `HepMC::IO_GenEvent::IO_GenEvent()`.

7.3.3.32 `std::ostream& HepMC::operator<< (std::ostream & ostr, const GenParticle & part)`

print particle

Definition at line 189 of file `GenParticle.cc`.

References `HepMC::GenVertex::barcode()`, `HepMC::GenParticle::barcode()`, `HepMC::FourVector::e()`, `HepMC::GenParticle::end_vertex()`, `HepMC::GenParticle::momentum()`, `HepMC::GenParticle::pdg_id()`, `HepMC::FourVector::px()`, `HepMC::FourVector::py()`, `HepMC::FourVector::pz()`, and `HepMC::GenParticle::status()`.

7.3.3.33 `std::ostream& HepMC::operator<< (std::ostream & ostr, const GenVertex & vtx)`

print vertex information

Definition at line 440 of file `GenVertex.cc`.

References `HepMC::GenVertex::barcode()`, `HepMC::GenVertex::position()`, and `HepMC::FourVector::x()`.

7.3.3.34 std::ostream& HepMC::operator<< (std::ostream & ostr, const ParticleData & pdata)

write to ostr

Definition at line 94 of file ParticleData.cc.

References HepMC::ParticleData::charge(), HepMC::ParticleData::clifetime(), HepMC::ParticleData::mass(), HepMC::ParticleData::name(), HepMC::ParticleData::pdg_id(), and HepMC::ParticleData::spin().

7.3.3.35 std::ostream& HepMC::operator<< (std::ostream & ostr, const Polarization & polar)

print polarization information

Definition at line 107 of file Polarization.cc.

References HepMC::Polarization::phi(), and HepMC::Polarization::theta().

7.3.4 Variable Documentation**7.3.4.1 const double HepMC::HepMC_hbarc [static]**

Initial value:

```
(6.6260755e-34 * (1.e-6/1.60217733e-19) / (2*3.14159265358979323846))
* (2.99792458e+8 * 1000.) * 1.e+3
```

hbar * c -> calculated with units of [mm*GeV]

Definition at line 56 of file ParticleData.h.

Referenced by clifetime_from_width(), HepMC::ParticleData::set_width(), and HepMC::ParticleData::width().

7.3.4.2 const double HepMC::HepMC_pi = 3.14159265358979323846 [static]

Definition at line 19 of file Polarization.h.

7.4 HepMC::detail Namespace Reference

Classes

- struct **enable_if**
internal - used to decide if a class is arithmetic
- struct **enable_if**< true, T >
internal - use if class T is arithmetic
- struct **disable_if**
internal - used by SimpleVector to decide if a class is arithmetic
- struct **disable_if**< false, T >
internal - used by SimpleVector to decide if a class is arithmetic
- struct **is_arithmetic**
undefined and therefore non-arithmetic
- struct **is_arithmetic**< char >
character is arithmetic
- struct **is_arithmetic**< unsigned char >
unsigned character is arithmetic
- struct **is_arithmetic**< signed char >
signed character is arithmetic
- struct **is_arithmetic**< short >
short is arithmetic
- struct **is_arithmetic**< unsigned short >
unsigned short is arithmetic
- struct **is_arithmetic**< int >
int is arithmetic
- struct **is_arithmetic**< unsigned int >
unsigned int is arithmetic
- struct **is_arithmetic**< long >
long is arithmetic
- struct **is_arithmetic**< unsigned long >
unsigned long is arithmetic
- struct **is_arithmetic**< float >
float is arithmetic
- struct **is_arithmetic**< double >

double is arithmetic

- **struct is_arithmetic< long double >**

long double is arithmetic

Functions

- **std::ostream & establish_output_stream_info (std::ostream &)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & establish_input_stream_info (std::istream &)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & read_vertex (std::istream &, TempParticleMap &, GenVertex *)**
- **std::istream & read_particle (std::istream &, TempParticleMap &, GenParticle *)**
- **std::ostream & output (std::ostream &os, const double &d)**
write a double - for internal use by streaming IO
- **std::ostream & output (std::ostream &os, const float &d)**
write a float - for internal use by streaming IO
- **std::ostream & output (std::ostream &os, const int &i)**
write an int - for internal use by streaming IO
- **std::ostream & output (std::ostream &os, const long &l)**
write a long - for internal use by streaming IO
- **std::ostream & output (std::ostream &os, const char &c)**
write a single char - for internal use by streaming IO
- **std::istream & find_event_end (std::istream &)**
used to read to the end of a bad event
- **std::istream & read_units (std::istream &is, GenEvent &evt)**

7.4.1 Function Documentation

7.4.1.1 std::ostream & HepMC::detail::establish_output_stream_info (std::ostream &)

used by **IO_GenEvent** (p. 164) constructor

Definition at line 649 of file GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), and HepMC::get_stream_info().

Referenced by HepMC::IO_GenEvent::IO_GenEvent().

7.4.1.2 std::istream & HepMC::detail::establish_input_stream_info (std::istream &)

used by **IO_GenEvent** (p. 164) constructor

Definition at line 663 of file GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), and HepMC::get_stream_info().

Referenced by HepMC::IO_GenEvent::IO_GenEvent().

7.4.1.3 std::istream & HepMC::detail::read_vertex (std::istream &, TempParticleMap &, GenVertex *)

get a **GenVertex** (p. 109) from ASCII input **TempParticleMap** (p. 235) is used to track the associations of particles with vertices

Definition at line 23 of file StreamHelpers.cc.

References find_event_end(), read_particle(), and v.

Referenced by HepMC::GenEvent::read().

7.4.1.4 std::istream & HepMC::detail::read_particle (std::istream &, TempParticleMap &, GenParticle *)

get a **GenParticle** (p. 99) from ASCII input **TempParticleMap** (p. 235) is used to track the associations of particles with vertices

Definition at line 541 of file GenEventStreamIO.cc.

References HepMC::TempParticleMap::addEndParticle(), HepMC::ascii, find_event_end(), HepMC::get_stream_info(), HepMC::StreamInfo::io_type(), p, and HepMC::Flow::set_icode().

Referenced by read_vertex().

7.4.1.5 std::ostream& HepMC::detail::output (std::ostream & os, const double & d) [inline]

write a double - for internal use by streaming IO

Definition at line 35 of file StreamHelpers.h.

Referenced by HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::operator<(), HepMC::GenEvent::write(), and HepMC::IO_AsciiParticles::write_event().

7.4.1.6 std::ostream& HepMC::detail::output (std::ostream & os, const float & d) [inline]

write a float - for internal use by streaming IO

Definition at line 47 of file StreamHelpers.h.

7.4.1.7 std::ostream& HepMC::detail::output (std::ostream & os, const int & i) [inline]

write an int - for internal use by streaming IO

Definition at line 59 of file StreamHelpers.h.

7.4.1.8 `std::ostream& HepMC::detail::output (std::ostream & os, const long & i)` `[inline]`

write a long - for internal use by streaming IO

Definition at line 71 of file StreamHelpers.h.

7.4.1.9 `std::ostream& HepMC::detail::output (std::ostream & os, const char & c)` `[inline]`

write a single char - for internal use by streaming IO

Definition at line 83 of file StreamHelpers.h.

7.4.1.10 `std::istream & HepMC::detail::find_event_end (std::istream &)`

used to read to the end of a bad event

Definition at line 98 of file StreamHelpers.cc.

Referenced by `HepMC::GenEvent::read()`, `read_particle()`, and `read_vertex()`.

7.4.1.11 `std::istream& HepMC::detail::read_units (std::istream & is, GenEvent & evt)`

Definition at line 622 of file GenEventStreamIO.cc.

References `HepMC::get_stream_info()`, `HepMC::StreamInfo::io_momentum_unit()`, `HepMC::StreamInfo::io_position_unit()`, and `HepMC::GenEvent::use_units()`.

7.5 HepMC::Units Namespace Reference

Enumerations

- enum **MomentumUnit** { MEV, GEV }
- enum **LengthUnit** { MM, CM }

Functions

- **LengthUnit default_length_unit ()**
default unit is defined by configure
- **MomentumUnit default_momentum_unit ()**
default unit is defined by configure
- **std::string name (MomentumUnit)**
convert enum to string
- **std::string name (LengthUnit)**
convert enum to string
- **double conversion_factor (MomentumUnit from, MomentumUnit to)**
scaling factor relative to MeV
- **double conversion_factor (LengthUnit from, LengthUnit to)**

7.5.1 Enumeration Type Documentation

7.5.1.1 enum HepMC::Units::MomentumUnit

Enumerator:

MEV

GEV

Definition at line 25 of file Units.h.

7.5.1.2 enum HepMC::Units::LengthUnit

Enumerator:

MM

CM

Definition at line 26 of file Units.h.

7.5.2 Function Documentation

7.5.2.1 LengthUnit HepMC::Units::default_length_unit ()

default unit is defined by configure

Examples:

`testUnits.cc.`

Referenced by HepMC::GenEvent::clear(), and main().

7.5.2.2 MomentumUnit HepMC::Units::default_momentum_unit ()

default unit is defined by configure

Examples:

`testUnits.cc.`

Referenced by HepMC::GenEvent::clear(), and main().

7.5.2.3 std::string HepMC::Units::name (MomentumUnit)

convert enum to string

Examples:

`testHepMC.cc.in, testStreamIO.cc.in, and testUnits.cc.`

Referenced by HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table(), main(), HepMC::IO_PDG_ParticleDataTable::read_entry(), HepMC::GenEvent::write(), and HepMC::GenEvent::write_units().

7.5.2.4 std::string HepMC::Units::name (LengthUnit)

convert enum to string

7.5.2.5 double HepMC::Units::conversion_factor (MomentumUnit *from*, MomentumUnit *to*)

scaling factor relative to MeV

Examples:

`testUnits.cc.`

Referenced by main().

7.5.2.6 double HepMC::Units::conversion_factor (LengthUnit *from*, LengthUnit *to*)

7.6 Units Namespace Reference

7.6.1 Detailed Description

Allow units to be specified within **HepMC** (p. 19). The default units are set at compile time.

Chapter 8

HepMC Class Documentation

8.1 HepMC::detail::disable_if<, > Struct Template Reference

internal - used by SimpleVector to decide if a class is arithmetic

```
#include <enable_if.h>
```

8.1.1 Detailed Description

```
template<bool, class> struct HepMC::detail::disable_if<, >
```

internal - used by SimpleVector to decide if a class is arithmetic

Definition at line 33 of file enable_if.h.

The documentation for this struct was generated from the following file:

- **enable_if.h**

8.2 HepMC::detail::disable_if< false, T > Struct Template Reference

internal - used by SimpleVector to decide if a class is arithmetic

```
#include <enable_if.h>
```

Public Types

- **typedef T type**
check type of class T

8.2.1 Detailed Description

template<class T> struct HepMC::detail::disable_if< false, T >

internal - used by SimpleVector to decide if a class is arithmetic

Definition at line 38 of file enable_if.h.

8.2.2 Member Typedef Documentation

8.2.2.1 template<class T> typedef T HepMC::detail::disable_if< false, T >::type

check type of class T

Definition at line 40 of file enable_if.h.

The documentation for this struct was generated from the following file:

- **enable_if.h**

8.3 HepMC::detail::enable_if<, > Struct Template Reference

internal - used to decide if a class is arithmetic

```
#include <enable_if.h>
```

8.3.1 Detailed Description

template<bool, class> struct HepMC::detail::enable_if<, >

internal - used to decide if a class is arithmetic

Definition at line 17 of file enable_if.h.

The documentation for this struct was generated from the following file:

- **enable_if.h**

8.4 HepMC::detail::enable_if< true, T > Struct Template Reference

internal - use if class T is arithmetic

```
#include <enable_if.h>
```

Public Types

- **typedef T type**
check type of class T

8.4.1 Detailed Description

```
template<class T> struct HepMC::detail::enable_if< true, T >
```

internal - use if class T is arithmetic

Definition at line 22 of file enable_if.h.

8.4.2 Member Typedef Documentation

8.4.2.1 template<class T> typedef T HepMC::detail::enable_if< true, T >::type

check type of class T

Definition at line 24 of file enable_if.h.

The documentation for this struct was generated from the following file:

- **enable_if.h**

8.5 HepMC::Flow Class Reference

The flow object.

```
#include <Flow.h>
```

Public Types

- **typedef std::map< int, int >::iterator iterator**
iterator for flow pattern container
- **typedef std::map< int, int >::const_iterator const_iterator**
const iterator for flow pattern container

Public Member Functions

- **Flow (GenParticle *particle_owner=0)**
default constructor
- **Flow (const Flow &)**
copy
- **virtual ~Flow ()**
- **void swap (Flow &other)**
swap
- **Flow & operator= (const Flow &)**
make a copy
- **bool operator== (const Flow &a) const**
equality
- **bool operator!= (const Flow &a) const**
inequality
- **void print (std::ostream &ostr=std::cout) const**
print Flow (p. 43) information to ostr
- **std::vector< HepMC::GenParticle * > connected_partners (int code, int code_index=1, int num_indices=2) const**
- **std::vector< HepMC::GenParticle * > dangling_connected_partners (int code, int code_index=1, int num_indices=2) const**
- **const GenParticle * particle_owner () const**
find particle owning this Flow (p. 43)
- **int icode (int code_index=1) const**
flow code
- **Flow set_icode (int code_index, int code)**

set flow code

- **Flow set_unique_icode (int code_index=1)**

set unique flow code

- **bool empty () const**

return true if there is no flow container

- **int size () const**

size of flow pattern container

- **void clear ()**

clear flow patterns

- **bool erase (int code_index)**

empty flow pattern container

- **iterator begin ()**

beginning of flow pattern container

- **iterator end ()**

end of flow pattern container

- **const_iterator begin () const**

beginning of flow pattern container

- **const_iterator end () const**

end of flow pattern container

Protected Member Functions

- **void connected_partners (std::vector< HepMC::GenParticle * > *output, int code, int code_index, int num_indices) const**

for internal use only

- **void dangling_connected_partners (std::vector< HepMC::GenParticle * > *output, std::vector< HepMC::GenParticle * > *visited_particles, int code, int code_index, int num_indices) const**

for internal use only

Friends

- **std::ostream & operator<< (std::ostream &ostr, const Flow &f)**

for printing

8.5.1 Detailed Description

The flow object.

The particle's flow object keeps track of an arbitrary number of flow patterns within a graph (i.e. color flow, charge flow, lepton number flow, ...) **Flow** (p. 43) patterns are coded with an integer, in the same manner as in Herwig.

Examples:

testFlow.cc.

Definition at line 66 of file Flow.h.

8.5.2 Member Typedef Documentation

8.5.2.1 `typedef std::map<int,int>::const_iterator HepMC::Flow::const_iterator`

const iterator for flow pattern container

Definition at line 128 of file Flow.h.

8.5.2.2 `typedef std::map<int,int>::iterator HepMC::Flow::iterator`

iterator for flow pattern container

Definition at line 126 of file Flow.h.

8.5.3 Constructor & Destructor Documentation

8.5.3.1 `HepMC::Flow::Flow (GenParticle * particle_owner = 0)`

default constructor

Definition at line 13 of file Flow.cc.

8.5.3.2 `HepMC::Flow::Flow (const Flow &)`

copy

copies both the m_icode AND the m_particle_owner

Definition at line 17 of file Flow.cc.

8.5.3.3 `HepMC::Flow::~~Flow ()` [virtual]

Definition at line 24 of file Flow.cc.

8.5.4 Member Function Documentation

8.5.4.1 `Flow::const_iterator HepMC::Flow::begin () const` [inline]

beginning of flow pattern container

Definition at line 186 of file Flow.h.

8.5.4.2 **Flow::iterator HepMC::Flow::begin ()** [inline]

beginning of flow pattern container

Definition at line 184 of file Flow.h.

8.5.4.3 **void HepMC::Flow::clear ()** [inline]

clear flow patterns

Definition at line 179 of file Flow.h.

8.5.4.4 **void HepMC::Flow::connected_partners (std::vector< HepMC::GenParticle * > * output, int code, int code_index, int num_indices) const** [protected]

for internal use only

protected: for recursive use by **Flow::connected_partners()** (p. 46)

Definition at line 60 of file Flow.cc.

References `HepMC::GenParticle::end_vertex()`, `HepMC::family`, `HepMC::not_in_vector()`, `p`, `HepMC::GenVertex::particles_begin()`, `HepMC::GenVertex::particles_end()`, and `HepMC::GenParticle::production_vertex()`.

8.5.4.5 **std::vector< GenParticle * > HepMC::Flow::connected_partners (int code, int code_index = 1, int num_indices = 2) const**

returns all connected particles which have "code" in any of the num_indices beginning with index code_index.

Returns all flow partners which have "code" in any of the num_indices beginning with index code_index. m_particle_owner is included in the result. Return is by value since the set should never be very big. EXAMPLE: if you want to find all flow partners that have the same code in indices 2,3,4 as particle p has in index 2, you would use: `set<GenParticle*> result = p->flow().connected_partners(p->flow().icode(2),2,3);`

Definition at line 38 of file Flow.cc.

References `icode()`, and `HepMC::detail::output()`.

8.5.4.6 **void HepMC::Flow::dangling_connected_partners (std::vector< HepMC::GenParticle * > * output, std::vector< HepMC::GenParticle * > * visited_particles, int code, int code_index, int num_indices) const** [protected]

for internal use only

protected: for recursive use by **Flow::dangling_connected_partners** (p. 47)

Definition at line 123 of file Flow.cc.

References `HepMC::GenParticle::end_vertex()`, `HepMC::family`, `HepMC::not_in_vector()`, `p`, `HepMC::GenVertex::particles_begin()`, `HepMC::GenVertex::particles_end()`, and `HepMC::GenParticle::production_vertex()`.

8.5.4.7 `std::vector< GenParticle * > HepMC::Flow::dangling_connected_partners (int code, int code_index = 1, int num_indices = 2) const`

same as `connected_partners`, but returns only those particles which are connected to ≤ 1 other particles (i.e. the flow line "dangles" at these particles)

Definition at line 108 of file `Flow.cc`.

References `icode()`, and `HepMC::detail::output()`.

8.5.4.8 `bool HepMC::Flow::empty () const` `[inline]`

return true if there is no flow container

Definition at line 177 of file `Flow.h`.

8.5.4.9 `Flow::const_iterator HepMC::Flow::end () const` `[inline]`

end of flow pattern container

Definition at line 187 of file `Flow.h`.

8.5.4.10 `Flow::iterator HepMC::Flow::end ()` `[inline]`

end of flow pattern container

Definition at line 185 of file `Flow.h`.

8.5.4.11 `bool HepMC::Flow::erase (int code_index)` `[inline]`

empty flow pattern container

Examples:

`testFlow.cc`.

Definition at line 180 of file `Flow.h`.

Referenced by `main()`.

8.5.4.12 `int HepMC::Flow::icode (int code_index = 1) const` `[inline]`

flow code

Definition at line 163 of file `Flow.h`.

Referenced by `connected_partners()`, `dangling_connected_partners()`, and `HepMC::GenParticle::flow()`.

8.5.4.13 `bool HepMC::Flow::operator!= (const Flow & a) const` `[inline]`

inequality

Definition at line 199 of file `Flow.h`.

8.5.4.14 Flow & HepMC::Flow::operator= (const Flow &) [inline]

make a copy

copies only the `m_icode` ... not the `particle_owner` this is intuitive behaviour so you can do `oneparticle->flow() = otherparticle->flow()`

Definition at line 202 of file `Flow.h`.

References `m_icode`.

8.5.4.15 bool HepMC::Flow::operator== (const Flow & a) const [inline]

equality

equivalent flows have the same flow codes for all `flow_numbers` (i.e. their `m_icode` maps are identical), but they need not have the same `m_particle` owner

Definition at line 193 of file `Flow.h`.

References `m_icode`.

8.5.4.16 const GenParticle * HepMC::Flow::particle_owner () const [inline]

find particle owning this **Flow** (p. 43)

Definition at line 160 of file `Flow.h`.

8.5.4.17 void HepMC::Flow::print (std::ostream & ostr = std::cout) const

print **Flow** (p. 43) information to `ostr`

Definition at line 34 of file `Flow.cc`.

8.5.4.18 Flow HepMC::Flow::set_icode (int code_index, int code) [inline]

set flow code

Definition at line 167 of file `Flow.h`.

Referenced by `HepMC::detail::read_particle()`, and `HepMC::GenParticle::set_flow()`.

8.5.4.19 Flow HepMC::Flow::set_unique_icode (int code_index = 1) [inline]

set unique flow code

use this method if you want to assign a unique flow code, but do not want the burden of choosing it yourself

Definition at line 171 of file `Flow.h`.

Referenced by `HepMC::GenParticle::set_flow()`.

8.5.4.20 int HepMC::Flow::size () const [inline]

size of flow pattern container

Definition at line 178 of file `Flow.h`.

8.5.4.21 void HepMC::Flow::swap (Flow & *other*)

swap

Definition at line 28 of file Flow.cc.

References `m_icode`, and `m_particle_owner`.

Referenced by `HepMC::GenParticle::swap()`.

8.5.5 Friends And Related Function Documentation

8.5.5.1 std::ostream& operator<< (std::ostream & *ostr*, const Flow & *f*) [friend]

for printing

Definition at line 190 of file Flow.cc.

The documentation for this class was generated from the following files:

- **Flow.h**
- **Flow.cc**

8.6 HepMC::FourVector Class Reference

FourVector (p. 50) is a simple representation of a physics 4 vector.

```
#include <SimpleVector.h>
```

Public Member Functions

- **FourVector (double xin, double yin, double zin, double tin=0)**
constructor requiring at least x, y, and z
- **FourVector (double t)**
constructor requiring only t
- **FourVector ()**
- **template<class T> FourVector (const T &v, typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type *=0)**
- **FourVector (const FourVector &v)**
copy constructor
- **void swap (FourVector &other)**
swap
- **double px () const**
return px
- **double py () const**
return py
- **double pz () const**
return pz
- **double e () const**
return E
- **double x () const**
return x
- **double y () const**
return y
- **double z () const**
return z
- **double t () const**
return t
- **double m2 () const**
Invariant mass squared.

- **double m () const**
Invariant mass. If $m2()$ (p. 53) is negative then $-\sqrt{-m2()}$ is returned.
- **double perp2 () const**
Transverse component of the spatial vector squared.
- **double perp () const**
Transverse component of the spatial vector (R in cylindrical system).
- **double theta () const**
The polar angle.
- **double phi () const**
The azimuth angle.
- **double rho () const**
spatial vector component magnitude
- **FourVector & operator= (const FourVector &)**
make a copy
- **bool operator== (const FourVector &) const**
equality
- **bool operator!= (const FourVector &) const**
inequality
- **double pseudoRapidity () const**
Returns the pseudo-rapidity, i.e. $-\ln(\tan(\theta/2))$.
- **double eta () const**
Pseudorapidity (of the space part).
- **void set (double x, double y, double z, double t)**
set x , y , z , and t
- **void setX (double x)**
set x
- **void setY (double y)**
set y
- **void setZ (double z)**
set z
- **void setT (double t)**
set t
- **void setPx (double x)**
set px

- **void setPy (double y)**

set py

- **void setPz (double z)**

set pz

- **void setE (double t)**

set E

8.6.1 Detailed Description

FourVector (p. 50) is a simple representation of a physics 4 vector.

For compatibility with existing code, the basic expected geometrical access methods are provided. Also, there is a templated constructor that will take another vector (`HepLorentzVector`, `GenVector`, ...) which must have the following methods: `x()` (p. 58), `y()` (p. 58), `z()` (p. 58), `t()` (p. 58).

Examples:

`testFlow.cc`, `testPrintBug.cc`, `testSimpleVector.cc`, and `VectorConversion.h`.

Definition at line 42 of file `SimpleVector.h`.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 `HepMC::FourVector::FourVector (double xin, double yin, double zin, double tin = 0)` `[inline]`

constructor requiring at least x, y, and z

Definition at line 47 of file `SimpleVector.h`.

8.6.2.2 `HepMC::FourVector::FourVector (double t)` `[inline]`

constructor requiring only t

Definition at line 51 of file `SimpleVector.h`.

8.6.2.3 `HepMC::FourVector::FourVector ()` `[inline]`

Definition at line 54 of file `SimpleVector.h`.

8.6.2.4 `template<class T> HepMC::FourVector::FourVector (const T & v, typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type * = 0)` `[inline]`

templated constructor this is used ONLY if T is not arithmetic

Definition at line 60 of file `SimpleVector.h`.

8.6.2.5 HepMC::FourVector::FourVector (const FourVector & v) [inline]

copy constructor

Definition at line 65 of file SimpleVector.h.

8.6.3 Member Function Documentation

8.6.3.1 double HepMC::FourVector::e () const [inline]

return E

Examples:

`testSimpleVector.cc.`

Definition at line 73 of file SimpleVector.h.

Referenced by `HepMC::GenParticle::convert_momentum()`, `main()`, `HepMC::operator<<()`, `HepMC::GenParticle::print()`, and `HepMC::IO_HEPEVT::write_event()`.

8.6.3.2 double HepMC::FourVector::eta () const

Pseudorapidity (of the space part).

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.3 double HepMC::FourVector::m () const

Invariant mass. If `m2()` (p. 53) is negative then `-sqrt(-m2())` is returned.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.4 double HepMC::FourVector::m2 () const

Invariant mass squared.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.5 bool HepMC::FourVector::operator!= (const FourVector &) const

inequality

8.6.3.6 FourVector& HepMC::FourVector::operator= (const FourVector &)

make a copy

8.6.3.7 bool HepMC::FourVector::operator== (const FourVector &) const

equality

8.6.3.8 double HepMC::FourVector::perp () const

Transverse component of the spatial vector (R in cylindrical system).

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.9 double HepMC::FourVector::perp2 () const

Transverse component of the spatial vector squared.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.10 double HepMC::FourVector::phi () const

The azimuth angle.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.11 double HepMC::FourVector::pseudoRapidity () const

Returns the pseudo-rapidity, i.e. $-\ln(\tan(\theta/2))$.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.6.3.12 double HepMC::FourVector::px () const [inline]

return px

Examples:

testSimpleVector.cc.

Definition at line 70 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), and HepMC::IO_HEPEVT::write_event().

8.6.3.13 double HepMC::FourVector::py () const [inline]

return py

Examples:

testSimpleVector.cc.

Definition at line 71 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), and HepMC::IO_HEPEVT::write_event().

8.6.3.14 double HepMC::FourVector::pz () const [inline]

return pz

Examples:

testSimpleVector.cc.

Definition at line 72 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), and HepMC::IO_HEPEVT::write_event().

8.6.3.15 double HepMC::FourVector::rho () const

spatial vector component magnitude

Examples:

testSimpleVector.cc.

Referenced by main().

8.6.3.16 void HepMC::FourVector::set (double x, double y, double z, double t)

set x, y, z, and t

Examples:

testSimpleVector.cc.

Referenced by main().

8.6.3.17 void HepMC::FourVector::setE (double *t*) [inline]

set E

Examples:

testSimpleVector.cc.

Definition at line 110 of file SimpleVector.h.

Referenced by main().

8.6.3.18 void HepMC::FourVector::setPx (double *x*) [inline]

set px

Examples:

testSimpleVector.cc.

Definition at line 107 of file SimpleVector.h.

Referenced by main().

8.6.3.19 void HepMC::FourVector::setPy (double *y*) [inline]

set py

Examples:

testSimpleVector.cc.

Definition at line 108 of file SimpleVector.h.

Referenced by main().

8.6.3.20 void HepMC::FourVector::setPz (double *z*) [inline]

set pz

Examples:

testSimpleVector.cc.

Definition at line 109 of file SimpleVector.h.

Referenced by main().

8.6.3.21 void HepMC::FourVector::setT (double *t*) [inline]

set t

Examples:

testSimpleVector.cc.

Definition at line 105 of file SimpleVector.h.

Referenced by main().

8.6.3.22 void HepMC::FourVector::setX (double *x*) [inline]

set x

Examples:

testSimpleVector.cc.

Definition at line 102 of file SimpleVector.h.

Referenced by main().

8.6.3.23 void HepMC::FourVector::setY (double *y*) [inline]

set y

Examples:

testSimpleVector.cc.

Definition at line 103 of file SimpleVector.h.

Referenced by main().

8.6.3.24 void HepMC::FourVector::setZ (double *z*) [inline]

set z

Examples:

testSimpleVector.cc.

Definition at line 104 of file SimpleVector.h.

Referenced by main().

8.6.3.25 void HepMC::FourVector::swap (FourVector & *other*)

swap

Referenced by HepMC::GenVertex::swap(), and HepMC::GenParticle::swap().

8.6.3.26 double HepMC::FourVector::t () const `[inline]`

return t

Examples:

testSimpleVector.cc.

Definition at line 78 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), main(), and HepMC::GenVertex::print().

8.6.3.27 double HepMC::FourVector::theta () const

The polar angle.

Examples:

testSimpleVector.cc.

Referenced by main().

8.6.3.28 double HepMC::FourVector::x () const `[inline]`

return x

Examples:

testSimpleVector.cc.

Definition at line 75 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), main(), HepMC::operator<<(), HepMC::GenVertex::point3d(), and HepMC::GenVertex::print().

8.6.3.29 double HepMC::FourVector::y () const `[inline]`

return y

Examples:

testSimpleVector.cc.

Definition at line 76 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), main(), HepMC::GenVertex::point3d(), and HepMC::GenVertex::print().

8.6.3.30 double HepMC::FourVector::z () const `[inline]`

return z

Examples:

testSimpleVector.cc.

Definition at line 77 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), main(), HepMC::GenVertex::point3d(), and HepMC::GenVertex::print().

The documentation for this class was generated from the following file:

- **SimpleVector.h**

8.7 HepMC::GenCrossSection Class Reference

The **GenCrossSection** (p. 60) class stores the generated cross section.

```
#include <GenCrossSection.h>
```

Public Member Functions

- **GenCrossSection ()**
- **~GenCrossSection ()**
- **GenCrossSection (GenCrossSection const &orig)**
copy
- **void swap (GenCrossSection &other)**
swap
- **GenCrossSection & operator= (GenCrossSection const &rhs)**
- **bool operator== (const GenCrossSection &) const**
check for equality
- **bool operator!= (const GenCrossSection &) const**
check for inequality
- **double cross_section () const**
cross section in pb
- **double cross_section_error () const**
error associated with this cross section in pb
- **bool is_set () const**
True if the cross section has been set. False by default.
- **void set_cross_section (double xs, double xs_err)**
Set cross section and error in pb.
- **void set_cross_section (double)**
set cross section in pb
- **void set_cross_section_error (double)**
set error associated with this cross section in pb
- **void clear ()**
- **std::ostream & write (std::ostream &) const**
write to an output stream
- **std::istream & read (std::istream &)**
read from an input stream

8.7.1 Detailed Description

The **GenCrossSection** (p. 60) class stores the generated cross section.

HepMC::GenCrossSection (p. 60) is used to store the generated cross section. This class is meant to be used to pass, on an event by event basis, the current best guess of the total cross section. It is expected that the final cross section will be stored elsewhere.

- double cross_section; // cross section in pb
- double cross_section_error; // error associated with this cross section

The units of cross_section and cross_section_error are expected to be pb.

GenCrossSection (p. 60) information will be written if **GenEvent** (p. 64) contains a pointer to a valid **GenCrossSection** (p. 60) object.

Examples:

testHepMC.cc.in.

Definition at line 32 of file GenCrossSection.h.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 HepMC::GenCrossSection::GenCrossSection () [inline]

Definition at line 35 of file GenCrossSection.h.

8.7.2.2 HepMC::GenCrossSection::~~GenCrossSection () [inline]

Definition at line 40 of file GenCrossSection.h.

8.7.2.3 HepMC::GenCrossSection::GenCrossSection (GenCrossSection const & orig)

copy

Definition at line 19 of file GenCrossSection.cc.

8.7.3 Member Function Documentation

8.7.3.1 void HepMC::GenCrossSection::clear ()

Clear all **GenCrossSection** (p. 60) info (disables output of **GenCrossSection** (p. 60) until the cross section is set again)

Definition at line 52 of file GenCrossSection.cc.

8.7.3.2 double HepMC::GenCrossSection::cross_section () const [inline]

cross section in pb

Definition at line 55 of file GenCrossSection.h.

Referenced by operator==(), and HepMC::GenEvent::write_cross_section().

8.7.3.3 double HepMC::GenCrossSection::cross_section_error () const [inline]

error associated with this cross section in pb

Definition at line 57 of file GenCrossSection.h.

Referenced by operator==(), and HepMC::GenEvent::write_cross_section().

8.7.3.4 bool HepMC::GenCrossSection::is_set () const [inline]

True if the cross section has been set. False by default.

Definition at line 60 of file GenCrossSection.h.

Referenced by write().

8.7.3.5 bool HepMC::GenCrossSection::operator!= (const GenCrossSection &) const

check for inequality

Definition at line 46 of file GenCrossSection.cc.

8.7.3.6 GenCrossSection & HepMC::GenCrossSection::operator= (GenCrossSection const & rhs)

shallow

Definition at line 32 of file GenCrossSection.cc.

References swap().

8.7.3.7 bool HepMC::GenCrossSection::operator== (const GenCrossSection &) const

check for equality

Definition at line 39 of file GenCrossSection.cc.

References cross_section(), and cross_section_error().

8.7.3.8 std::istream & HepMC::GenCrossSection::read (std::istream &)

read from an input stream

Definition at line 76 of file GenCrossSection.cc.

References set_cross_section().

Referenced by HepMC::operator>>().

8.7.3.9 void HepMC::GenCrossSection::set_cross_section (double) [inline]

set cross section in pb

Definition at line 103 of file GenCrossSection.h.

8.7.3.10 void HepMC::GenCrossSection::set_cross_section (double *xs*, double *xs_err*) [inline]

Set cross section and error in pb.

Examples:

testHepMC.cc.in.

Definition at line 98 of file GenCrossSection.h.

References set_cross_section_error().

Referenced by read().

8.7.3.11 void HepMC::GenCrossSection::set_cross_section_error (double) [inline]

set error associated with this cross section in pb

Definition at line 109 of file GenCrossSection.h.

Referenced by set_cross_section().

8.7.3.12 void HepMC::GenCrossSection::swap (GenCrossSection & *other*)

swap

Definition at line 25 of file GenCrossSection.cc.

References m_cross_section, m_cross_section_error, and m_is_set.

Referenced by operator=().

8.7.3.13 std::ostream & HepMC::GenCrossSection::write (std::ostream &) const

write to an output stream

Definition at line 59 of file GenCrossSection.cc.

References is_set().

Referenced by HepMC::operator<<(), and HepMC::GenEvent::write().

The documentation for this class was generated from the following files:

- **GenCrossSection.h**
- **GenCrossSection.cc**

8.8 HepMC::GenEvent Class Reference

The **GenEvent** (p. 64) class is the core of **HepMC** (p. 19).

```
#include <GenEvent.h>
```

Public Member Functions

- **GenEvent** (int signal_process_id=0, int event_number=0, GenVertex *signal_vertex=0, const WeightContainer &weights=std::vector< double >(), const std::vector< long > &randomstates=std::vector< long >(), Units::MomentumUnit=Units::default_momentum_unit(), Units::LengthUnit=Units::default_length_unit())
default constructor creates null pointers to HeavyIon (p.133), PdfInfo (p.218), and GenCrossSection (p.60)
- **GenEvent** (int signal_process_id, int event_number, GenVertex *signal_vertex, const WeightContainer &weights, const std::vector< long > &randomstates, const HeavyIon &ion, const PdfInfo &pdf, Units::MomentumUnit=Units::default_momentum_unit(), Units::LengthUnit=Units::default_length_unit())
explicit constructor that takes HeavyIon (p.133) and PdfInfo (p.218)
- **GenEvent** (Units::MomentumUnit, Units::LengthUnit, int signal_process_id=0, int event_number=0, GenVertex *signal_vertex=0, const WeightContainer &weights=std::vector< double >(), const std::vector< long > &randomstates=std::vector< long >())
constructor requiring units - all else is default
- **GenEvent** (Units::MomentumUnit, Units::LengthUnit, int signal_process_id, int event_number, GenVertex *signal_vertex, const WeightContainer &weights, const std::vector< long > &randomstates, const HeavyIon &ion, const PdfInfo &pdf)
explicit constructor with units first that takes HeavyIon (p.133) and PdfInfo (p.218)
- **GenEvent** (const GenEvent &inevent)
deep copy
- **GenEvent & operator=** (const GenEvent &inevent)
make a deep copy
- **virtual ~GenEvent** ()
deletes all vertices/particles in this evt
- **void swap** (GenEvent &other)
swap
- **void print** (std::ostream &ostr=std::cout) const
dumps to ostr
- **void print_version** (std::ostream &ostr=std::cout) const
dumps release version to ostr
- **GenParticle * barcode_to_particle** (int barCode) const
assign a barcode to a particle

- **GenVertex * barcode_to_vertex (int barCode) const**
assign a barcode to a vertex
- **int signal_process_id () const**
unique signal process id
- **int event_number () const**
event number
- **int mpi () const**
number of multi parton interactions
- **double event_scale () const**
energy scale, see hep-ph/0109068
- **double alphaQCD () const**
QCD coupling, see hep-ph/0109068.
- **double alphaQED () const**
- **GenVertex * signal_process_vertex () const**
pointer to the vertex containing the signal process
- **bool valid_beam_particles () const**
test to see if we have two valid beam particles
- **std::pair< HepMC::GenParticle *, HepMC::GenParticle * > beam_particles () const**
pair of pointers to the two incoming beam particles
- **bool is_valid () const**
- **WeightContainer & weights ()**
direct access to WeightContainer (p. 244)
- **const WeightContainer & weights () const**
direct access to WeightContainer (p. 244)
- **GenCrossSection const * cross_section () const**
access the GenCrossSection (p. 60) container if it exists
- **GenCrossSection * cross_section ()**
- **HeavyIon const * heavy_ion () const**
access the HeavyIon (p. 133) container if it exists
- **HeavyIon * heavy_ion ()**
- **PdfInfo const * pdf_info () const**
access the PdfInfo (p. 218) container if it exists
- **PdfInfo * pdf_info ()**
- **const std::vector< long > & random_states () const**
vector of integers containing information about the random state

- **int particles_size () const**
how many particle barcodes exist?
- **bool particles_empty () const**
return true if there are no particle barcodes
- **int vertices_size () const**
how many vertex barcodes exist?
- **bool vertices_empty () const**
return true if there are no vertex barcodes
- **void write_units (std::ostream &os=std::cout) const**
- **void write_cross_section (std::ostream &ostr=std::cout) const**
- **Units::MomentumUnit momentum_unit () const**
Units (p. 35) used by the GenParticle (p. 99) momentum FourVector (p. 50).
- **Units::LengthUnit length_unit () const**
Units (p. 35) used by the GenVertex (p. 109) position FourVector (p. 50).
- **std::ostream & write (std::ostream &)**
- **std::istream & read (std::istream &)**
- **bool add_vertex (GenVertex *vtx)**
adds to evt and adopts
- **bool remove_vertex (GenVertex *vtx)**
erases vtx from evt
- **void clear ()**
empties the entire event
- **void set_signal_process_id (int id)**
set unique signal process id
- **void set_event_number (int eventno)**
set event number
- **void set_mpi (int)**
Use this to set the number of multi parton interactions in each event.
- **void set_event_scale (double scale)**
set energy scale
- **void set_alphaQCD (double a)**
set QCD coupling
- **void set_alphaQED (double a)**
set QED coupling

- **void set_signal_process_vertex (GenVertex *)**
set pointer to the vertex containing the signal process
- **bool set_beam_particles (GenParticle *, GenParticle *)**
set incoming beam particles
- **bool set_beam_particles (std::pair< HepMC::GenParticle *, HepMC::GenParticle * > const &)**
use a pair of GenParticle's to set incoming beam particles*
- **void set_random_states (const std::vector< long > &randomstates)**
provide random state information
- **void set_cross_section (const GenCrossSection &)**
provide a pointer to the GenCrossSection (p. 60) container
- **void set_heavy_ion (const HeavyIon &ion)**
provide a pointer to the HeavyIon (p. 133) container
- **void set_pdf_info (const PdfInfo &p)**
provide a pointer to the PdfInfo (p. 218) container
- **void use_units (Units::MomentumUnit, Units::LengthUnit)**
set the units using enums
- **void use_units (std::string &, std::string &)**
- **vertex_const_iterator vertices_begin () const**
begin vertex iteration
- **vertex_const_iterator vertices_end () const**
end vertex iteration
- **vertex_iterator vertices_begin ()**
begin vertex iteration
- **vertex_iterator vertices_end ()**
end vertex iteration
- **particle_const_iterator particles_begin () const**
begin particle iteration
- **particle_const_iterator particles_end () const**
end particle iteration
- **particle_iterator particles_begin ()**
begin particle iteration
- **particle_iterator particles_end ()**
end particle iteration

Protected Member Functions

- **bool set_barcode (GenParticle *p, int suggested_barcode=false)**
set the barcode - intended for use by GenParticle (p. 99)
- **bool set_barcode (GenVertex *v, int suggested_barcode=false)**
set the barcode - intended for use by GenVertex (p. 109)
- **void remove_barcode (GenParticle *p)**
intended for use by GenParticle (p. 99)
- **void remove_barcode (GenVertex *v)**
intended for use by GenVertex (p. 109)
- **void delete_all_vertices ()**
delete all vertices owned by this event

Friends

- class **GenParticle**
- class **GenVertex**
- class **vertex_const_iterator**
- class **vertex_iterator**
- class **particle_const_iterator**
- class **particle_iterator**

Classes

- class **particle_const_iterator**
const particle iterator
- class **particle_iterator**
non-const particle iterator
- class **vertex_const_iterator**
const vertex iterator
- class **vertex_iterator**
non-const vertex iterator

8.8.1 Detailed Description

The **GenEvent** (p. 64) class is the core of **HepMC** (p. 19).

HepMC::GenEvent (p. 64) contains information about generated particles. **GenEvent** (p. 64) is structured as a set of vertices which contain the particles.

Examples:

example_BuildEventFromScratch.cc, example_EventSelection.cc, example_MyHerwig.cc, example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc, example_UsingIterators.cc, testFlow.cc, testHepMC.cc.in, testHepMCIteration.cc.in, testHerwigCopies.cc, testMass.cc.in, testMultipleCopies.cc.in, testPrintBug.cc, testPythiaCopies.cc, and testStreamIO.cc.in.

Definition at line 150 of file GenEvent.h.

8.8.2 Constructor & Destructor Documentation

8.8.2.1 `HepMC::GenEvent::GenEvent (int signal_process_id = 0, int event_number = 0, GenVertex * signal_vertex = 0, const WeightContainer & weights = std::vector< double >(), const std::vector< long > & randomstates = std::vector< long >(), Units::MomentumUnit = Units::default_momentum_unit(), Units::LengthUnit = Units::default_length_unit())`

default constructor creates null pointers to **HeavyIon** (p. 133), **PdfInfo** (p. 218), and **GenCrossSection** (p. 60)

This constructor only allows null pointers to **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 22 of file GenEvent.cc.

8.8.2.2 `HepMC::GenEvent::GenEvent (int signal_process_id, int event_number, GenVertex * signal_vertex, const WeightContainer & weights, const std::vector< long > & randomstates, const HeavyIon & ion, const PdfInfo & pdf, Units::MomentumUnit = Units::default_momentum_unit(), Units::LengthUnit = Units::default_length_unit())`

explicit constructor that takes **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

GenEvent (p. 64) makes its own copy of **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 55 of file GenEvent.cc.

8.8.2.3 `HepMC::GenEvent::GenEvent (Units::MomentumUnit, Units::LengthUnit, int signal_process_id = 0, int event_number = 0, GenVertex * signal_vertex = 0, const WeightContainer & weights = std::vector< double >(), const std::vector< long > & randomstates = std::vector< long >())`

constructor requiring units - all else is default

constructor requiring units - all else is default This constructor only allows null pointers to **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 88 of file GenEvent.cc.

8.8.2.4 HepMC::GenEvent::GenEvent (Units::MomentumUnit, Units::LengthUnit, int *signal_process_id*, int *event_number*, GenVertex * *signal_vertex*, const WeightContainer & *weights*, const std::vector< long > & *randomstates*, const HeavyIon & *ion*, const PdfInfo & *pdf*)

explicit constructor with units first that takes **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

explicit constructor with units first that takes **HeavyIon** (p. 133) and **PdfInfo** (p. 218) **GenEvent** (p. 64) makes its own copy of **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

note: default values for *m_event_scale*, *m_alphaQCD*, *m_alphaQED* are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 122 of file GenEvent.cc.

8.8.2.5 HepMC::GenEvent::GenEvent (const GenEvent & *inevent*)

deep copy

deep copy - makes a copy of all vertices!

Definition at line 156 of file GenEvent.cc.

References *add_vertex()*, *beam_particles()*, *GenParticle*, *GenVertex*, *p*, *particles_begin()*, *particles_end()*, *random_states()*, *set_beam_particles()*, *set_random_states()*, *set_signal_process_vertex()*, *signal_process_vertex()*, *v*, *vertices_begin()*, *vertices_end()*, and *weights()*.

8.8.2.6 HepMC::GenEvent::~~GenEvent () [virtual]

deletes all vertices/particles in this evt

Deep destructor. deletes all vertices/particles in this **GenEvent** (p. 64) deletes the associated **HeavyIon** (p. 133) and **PdfInfo** (p. 218)

Definition at line 258 of file GenEvent.cc.

References *delete_all_vertices()*.

8.8.3 Member Function Documentation

8.8.3.1 bool HepMC::GenEvent::add_vertex (GenVertex * *vtx*)

adds to evt and adopts

returns true if successful - generally will only return false if the inserted vertex is already included in the event.

Examples:

example_BuildEventFromScratch.cc, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 336 of file GenEvent.cc.

References *HepMC::GenVertex::barcode()*, *HepMC::GenVertex::parent_event()*, *remove_vertex()*, and *HepMC::GenVertex::set_parent_event_()*.

Referenced by *HepMC::IO_HERWIG::build_end_vertex()*, *HepMC::IO_HEPEVT::build_end_vertex()*, *HepMC::IO_HERWIG::build_production_vertex()*, *HepMC::IO_HEPEVT::build_production_vertex()*,

HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), GenEvent(), main(), read(), and set_signal_process_vertex().

8.8.3.2 double HepMC::GenEvent::alphaQCD () const [inline]

QCD coupling, see hep-ph/0109068.

Definition at line 662 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.3 double HepMC::GenEvent::alphaQED () const [inline]

QED coupling, see hep-ph/0109068

Definition at line 664 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.4 GenParticle * HepMC::GenEvent::barcode_to_particle (int *barCode*) const [inline]

assign a barcode to a particle

Each vertex or particle has a barcode, which is just an integer which uniquely identifies it inside the event (i.e. there is a one to one mapping between particle memory addresses and particle barcodes... and the same applied for vertices).

The value of a barcode has NO MEANING and NO ORDER! For the user's convenience, when an event is read in via an IO_method from an indexed list (like the HEPEVT common block), then the index will become the barcode for that particle.

Particle barcodes are always positive integers. The barcodes are chosen and set automatically when a vertex or particle comes under the ownership of an event (i.e. it is contained in an event).

Please note that the barcodes are intended for internal use within **HepMC** (p. 19) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Definition at line 770 of file GenEvent.h.

8.8.3.5 GenVertex * HepMC::GenEvent::barcode_to_vertex (int *barCode*) const [inline]

assign a barcode to a vertex

Each vertex or particle has a barcode, which is just an integer which uniquely identifies it inside the event (i.e. there is a one to one mapping between particle memory addresses and particle barcodes... and the same applied for vertices).

The value of a barcode has NO MEANING and NO ORDER! For the user's convenience, when an event is read in via an IO_method from an indexed list (like the HEPEVT common block), then the index will become the barcode for that particle.

Vertex barcodes are always negative integers. The barcodes are chosen and set automatically when a vertex or particle comes under the ownership of an event (i.e. it is contained in an event).

Please note that the barcodes are intended for internal use within **HepMC** (p. 19) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Definition at line 795 of file GenEvent.h.

Referenced by HepMC::compareVertices(), and read().

8.8.3.6 `std::pair< HepMC::GenParticle *, HepMC::GenParticle * >
HepMC::GenEvent::beam_particles () const [inline]`

pair of pointers to the two incoming beam particles

Examples:

`testMass.cc.in.`

Definition at line 816 of file GenEvent.h.

Referenced by HepMC::compareBeamParticles(), GenEvent(), print(), and write().

8.8.3.7 `void HepMC::GenEvent::clear ()`

empties the entire event

remove all information from the event deletes all vertices/particles in this evt

Examples:

`testHepMCIteration.cc.in, and testStreamIO.cc.in.`

Definition at line 367 of file GenEvent.cc.

References HepMC::Units::default_length_unit(), HepMC::Units::default_momentum_unit(), and delete_all_vertices().

Referenced by HepMC::IO_GenEvent::fill_next_event(), and read().

8.8.3.8 `GenCrossSection * HepMC::GenEvent::cross_section () [inline]`

Definition at line 679 of file GenEvent.h.

8.8.3.9 `GenCrossSection const * HepMC::GenEvent::cross_section () const [inline]`

access the **GenCrossSection** (p. 60) container if it exists

Examples:

`example_PythiaStreamIO.cc, and testHepMC.cc.in.`

Definition at line 676 of file GenEvent.h.

Referenced by readPythiaStreamIO(), and write_cross_section().

8.8.3.10 void HepMC::GenEvent::delete_all_vertices () [protected]

delete all vertices owned by this event

deletes all vertices in the vertex container (i.e. all vertices owned by this event) The vertices are the "owners" of the particles, so as we delete the vertices, the vertex destructors are automatically deleting their particles.

Definition at line 405 of file GenEvent.cc.

References particles_empty(), and vertices_empty().

Referenced by clear(), and ~GenEvent().

8.8.3.11 int HepMC::GenEvent::event_number () const [inline]

event number

Examples:

example_EventSelection.cc, example_MyPythia.cc, testHepMC.cc.in, testHepMCIteration.cc.in, testHerwigCopies.cc, testMass.cc.in, testMultipleCopies.cc.in, testPythiaCopies.cc, and test-StreamIO.cc.in.

Definition at line 654 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), main(), particleTypes(), print(), pythia_in(), pythia_in_out(), read(), write(), HepMC::IO_HEPEVT::write_event(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.12 double HepMC::GenEvent::event_scale () const [inline]

energy scale, see hep-ph/0109068

Definition at line 660 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.13 HeavyIon * HepMC::GenEvent::heavy_ion () [inline]

Definition at line 685 of file GenEvent.h.

8.8.3.14 HeavyIon const * HepMC::GenEvent::heavy_ion () const [inline]

access the **HeavyIon** (p. 133) container if it exists

Definition at line 682 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), and write().

8.8.3.15 bool HepMC::GenEvent::is_valid () const

check **GenEvent** (p. 64) for validity A **GenEvent** (p. 64) is presumed valid if it has particles and/or vertices.

A **GenEvent** (p. 64) is presumed valid if it has both associated particles and vertices. No other information is checked.

Examples:

example_PythiaStreamIO.cc, and testStreamIO.cc.in.

Definition at line 667 of file GenEvent.cc.

References particles_empty(), and vertices_empty().

Referenced by HepMC::IO_GenEvent::fill_next_event(), and readPythiaStreamIO().

8.8.3.16 Units::LengthUnit HepMC::GenEvent::length_unit () const [inline]

Units (p. 35) used by the **GenVertex** (p. 109) position **FourVector** (p. 50).

Definition at line 824 of file GenEvent.h.

Referenced by write(), and write_units().

8.8.3.17 Units::MomentumUnit HepMC::GenEvent::momentum_unit () const [inline]

Units (p. 35) used by the **GenParticle** (p. 99) momentum **FourVector** (p. 50).

Definition at line 821 of file GenEvent.h.

Referenced by write(), and write_units().

8.8.3.18 int HepMC::GenEvent::mpi () const [inline]

number of multi parton interactions

Returns the number of multi parton interactions in the event. This number is -1 if it is not set.

Definition at line 658 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), and write().

8.8.3.19 GenEvent & HepMC::GenEvent::operator= (const GenEvent & inevent)

make a deep copy

best practices implementation

Definition at line 269 of file GenEvent.cc.

References swap().

8.8.3.20 particle_iterator HepMC::GenEvent::particles_begin () [inline]

begin particle iteration

Definition at line 541 of file GenEvent.h.

8.8.3.21 particle_const_iterator HepMC::GenEvent::particles_begin () const [inline]

begin particle iteration

Examples:

example_BuildEventFromScratch.cc, example_EventSelection.cc, example_MyPythia.cc, example_UsingIterators.cc, testHepMCIteration.cc.in, testMass.cc.in, and testMultipleCopies.cc.in.

Definition at line 483 of file GenEvent.h.

Referenced by HepMC::compareParticles(), findPiZero(), GenEvent(), main(), IsGoodEvent::operator>(), IsGoodEventMyPythia::operator>(), IsEventGood::operator>(), particleTypes(), valid_beam_particles(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.22 bool HepMC::GenEvent::particles_empty () const [inline]

return true if there are no particle barcodes

Definition at line 805 of file GenEvent.h.

Referenced by delete_all_vertices(), and is_valid().

8.8.3.23 particle_iterator HepMC::GenEvent::particles_end () [inline]

end particle iteration

Definition at line 545 of file GenEvent.h.

8.8.3.24 particle_const_iterator HepMC::GenEvent::particles_end () const [inline]

end particle iteration

Examples:

example_BuildEventFromScratch.cc, example_EventSelection.cc, example_MyPythia.cc, example_UsingIterators.cc, testHepMCIteration.cc.in, testMass.cc.in, and testMultipleCopies.cc.in.

Definition at line 487 of file GenEvent.h.

Referenced by HepMC::compareParticles(), findPiZero(), GenEvent(), main(), IsGoodEvent::operator>(), IsGoodEventMyPythia::operator>(), IsEventGood::operator>(), particleTypes(), valid_beam_particles(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.25 int HepMC::GenEvent::particles_size () const [inline]

how many particle barcodes exist?

Examples:

testMultipleCopies.cc.in.

Definition at line 802 of file GenEvent.h.

Referenced by HepMC::compareParticles(), particleTypes(), print(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.26 PdfInfo * HepMC::GenEvent::pdf_info () [inline]

Definition at line 691 of file GenEvent.h.

8.8.3.27 PdfInfo const * HepMC::GenEvent::pdf_info () const [inline]

access the **PdfInfo** (p. 218) container if it exists

Definition at line 688 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), and write().

8.8.3.28 void HepMC::GenEvent::print (std::ostream & ostr = std::cout) const

dumps to ostr

dumps the content of this event to ostr to dump to cout use: event.print(); if you want to write this event to file outfile.txt you could use: std::ofstream outfile("outfile.txt"); event.print(outfile);

Examples:

example_BuildEventFromScratch.cc, example_MyHerwig.cc, testFlow.cc, testHepMC.cc.in, testHerwigCopies.cc, testMultipleCopies.cc.in, testPrintBug.cc, and testPythiaCopies.cc.

Definition at line 277 of file GenEvent.cc.

References alphaQCD(), alphaQED(), HepMC::GenVertex::barcode(), beam_particles(), HepMC::WeightContainer::end(), event_number(), event_scale(), particles_size(), signal_process_id(), signal_process_vertex(), HepMC::WeightContainer::size(), vertices_end(), vertices_size(), weights(), write_cross_section(), and write_units().

Referenced by main().

8.8.3.29 void HepMC::GenEvent::print_version (std::ostream & ostr = std::cout) const

dumps release version to ostr

Definition at line 330 of file GenEvent.cc.

References HepMC::writeVersion().

8.8.3.30 const std::vector< long > & HepMC::GenEvent::random_states () const [inline]

vector of integers containing information about the random state

Vector of integers which specify the random number generator's state for this event. It is left to the generator to make use of this. We envision a vector of RndmStatesTags to be included with a run class which would specify the meaning of the random_states.

Definition at line 699 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), GenEvent(), read(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.31 std::istream & HepMC::GenEvent::read (std::istream &)

Examples:

example_PythiaStreamIO.cc, and testStreamIO.cc.in.

Definition at line 138 of file GenEventStreamIO.cc.

References HepMC::GenVertex::add_particle_in(), add_vertex(), barcode_to_vertex(), clear(), HepMC::TempParticleMap::end_vertex(), event_number(), HepMC::extascii, HepMC::detail::find_event_end(), HepMC::StreamInfo::finished_first_event(), HepMC::gen, GenVertex, HepMC::get_stream_info(), HepMC::StreamInfo::has_key(), HepMC::StreamInfo::io_type(), HepMC::PdfInfo::is_valid(), HepMC::TempParticleMap::order_begin(), HepMC::TempParticleMap::order_end(), p, random_states(), HepMC::detail::read_vertex(), set_alphaQCD(), set_alphaQED(), set_beam_particles(), set_cross_section(), set_event_number(), set_event_scale(), HepMC::StreamInfo::set_finished_first_event(), set_heavy_ion(), set_mpi(), set_pdf_info(), set_random_states(), set_signal_process_id(), set_signal_process_vertex(), signal_process_id(), signal_process_vertex(), v, and weights().

Referenced by HepMC::operator>>(), and readPythiaStreamIO().

8.8.3.32 void HepMC::GenEvent::remove_barcode (GenVertex * v) [inline, protected]

intended for use by **GenVertex** (p. 109)

Definition at line 749 of file GenEvent.h.

References v.

8.8.3.33 void HepMC::GenEvent::remove_barcode (GenParticle * p) [inline, protected]

intended for use by **GenParticle** (p. 99)

Definition at line 746 of file GenEvent.h.

References p.

Referenced by HepMC::GenParticle::set_end_vertex_(), HepMC::GenVertex::set_parent_event_(), HepMC::GenParticle::set_production_vertex_(), HepMC::GenParticle::~~GenParticle(), and HepMC::GenVertex::~~GenVertex().

8.8.3.34 bool HepMC::GenEvent::remove_vertex (GenVertex * vtx)

erases vtx from evt

this removes vtx from the event but does NOT delete it. returns True if an entry vtx existed in the table and was erased

Definition at line 359 of file GenEvent.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::parent_event(), and HepMC::GenVertex::set_parent_event_().

Referenced by add_vertex().

8.8.3.35 void HepMC::GenEvent::set_alphaQCD (double a) [inline]

set QCD coupling

Definition at line 715 of file GenEvent.h.

Referenced by read().

8.8.3.36 void HepMC::GenEvent::set_alphaQED (double *a*) [inline]

set QED coupling

Definition at line 717 of file GenEvent.h.

Referenced by read().

8.8.3.37 bool HepMC::GenEvent::set_barcode (GenVertex * *v*, int *suggested_barcode* = false) [protected]

set the barcode - intended for use by **GenVertex** (p. 109)

Definition at line 503 of file GenEvent.cc.

References v.

8.8.3.38 bool HepMC::GenEvent::set_barcode (GenParticle * *p*, int *suggested_barcode* = false) [protected]

set the barcode - intended for use by **GenParticle** (p. 99)

Definition at line 432 of file GenEvent.cc.

References p.

Referenced by HepMC::GenVertex::set_parent_event(), HepMC::GenVertex::suggest_barcode(), and HepMC::GenParticle::suggest_barcode().

8.8.3.39 bool HepMC::GenEvent::set_beam_particles (std::pair< HepMC::GenParticle *, HepMC::GenParticle * > const & *bp*)

use a pair of GenParticle*'s to set incoming beam particles

construct the beam particle information using a std::pair of pointers to **GenParticle** (p. 99) returns false if either GenParticle* is null

Definition at line 597 of file GenEvent.cc.

References set_beam_particles().

8.8.3.40 bool HepMC::GenEvent::set_beam_particles (GenParticle * *bp1*, GenParticle * *bp2*)

set incoming beam particles

construct the beam particle information using pointers to **GenParticle** (p. 99) returns false if either GenParticle* is null

Definition at line 588 of file GenEvent.cc.

Referenced by HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), GenEvent(), read(), and set_beam_particles().

8.8.3.41 void HepMC::GenEvent::set_cross_section (const GenCrossSection &) [inline]

provide a pointer to the **GenCrossSection** (p. 60) container

Examples:

example_MyHerwig.cc, example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc, testHepMC.cc.in, testHerwigCopies.cc, and testPythiaCopies.cc.

Definition at line 724 of file GenEvent.h.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), read(), and writePythiaStreamIO().

8.8.3.42 void HepMC::GenEvent::set_event_number (int eventno) [inline]

set event number

Examples:

example_MyHerwig.cc, example_MyPythia.cc, example_PythiaStreamIO.cc, and testHerwigCopies.cc.

Definition at line 705 of file GenEvent.h.

Referenced by HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), read(), and writePythiaStreamIO().

8.8.3.43 void HepMC::GenEvent::set_event_scale (double scale) [inline]

set energy scale

Definition at line 713 of file GenEvent.h.

Referenced by read().

8.8.3.44 void HepMC::GenEvent::set_heavy_ion (const HeavyIon & ion) [inline]

provide a pointer to the **HeavyIon** (p. 133) container

Examples:

testMass.cc.in.

Definition at line 730 of file GenEvent.h.

Referenced by read().

8.8.3.45 void HepMC::GenEvent::set_mpi (int) [inline]

Use this to set the number of multi parton interactions in each event.

Examples:

example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc, and testPythiaCopies.cc.

Definition at line 709 of file GenEvent.h.

Referenced by event_selection(), main(), pythia_out(), read(), and writePythiaStreamIO().

8.8.3.46 void HepMC::GenEvent::set_pdf_info (const PdfInfo & p) [inline]

provide a pointer to the **PdfInfo** (p. 218) container

Examples:

testMass.cc.in.

Definition at line 736 of file GenEvent.h.

References p.

Referenced by read().

8.8.3.47 void HepMC::GenEvent::set_random_states (const std::vector< long > & randomstates) [inline]

provide random state information

Definition at line 742 of file GenEvent.h.

Referenced by GenEvent(), and read().

8.8.3.48 void HepMC::GenEvent::set_signal_process_id (int id) [inline]

set unique signal process id

Examples:

example_MyHerwig.cc, example_MyPythia.cc, example_PythiaStreamIO.cc, and testHerwig-Copies.cc.

Definition at line 702 of file GenEvent.h.

Referenced by main(), pythia_in_out(), pythia_out(), pythia_particle_out(), read(), and writePythiaStreamIO().

8.8.3.49 void HepMC::GenEvent::set_signal_process_vertex (GenVertex *) [inline]

set pointer to the vertex containing the signal process

Examples:

example_BuildEventFromScratch.cc, and testFlow.cc.

Definition at line 719 of file GenEvent.h.

References add_vertex().

Referenced by HepMC::IO_HERWIG::fill_next_event(), GenEvent(), main(), and read().

8.8.3.50 int HepMC::GenEvent::signal_process_id () const [inline]

unique signal process id

The integer ID that uniquely specifies this signal process, i.e. MSUB in Pythia. It is necessary to package this with each event rather than with the run because many processes may be generated within one run.

Definition at line 651 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), read(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.51 GenVertex * HepMC::GenEvent::signal_process_vertex () const [inline]

pointer to the vertex containing the signal process

returns a (mutable) pointer to the signal process vertex

Definition at line 666 of file GenEvent.h.

Referenced by HepMC::compareSignalProcessVertex(), GenEvent(), print(), read(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.52 void HepMC::GenEvent::swap (GenEvent & other)

swap

Definition at line 226 of file GenEvent.cc.

References m_alphaQCD, m_alphaQED, m_beam_particle_1, m_beam_particle_2, m_cross_section, m_event_number, m_event_scale, m_heavy_ion, m_momentum_unit, m_mpi, m_particle_barcodes, m_pdf_info, m_position_unit, m_random_states, m_signal_process_id, m_signal_process_vertex, m_vertex_barcodes, m_weights, HepMC::WeightContainer::swap(), vertices_begin(), and vertices_end().

Referenced by operator=().

8.8.3.53 void HepMC::GenEvent::use_units (std::string &, std::string &) [inline]

set the units using strings the string must match the enum exactly

Definition at line 833 of file GenEvent.h.

8.8.3.54 void HepMC::GenEvent::use_units (Units::MomentumUnit, Units::LengthUnit) [inline]

set the units using enums

Examples:

example_BuildEventFromScratch.cc, example_MyHerwig.cc, example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc, testHerwigCopies.cc, and testPythiaCopies.cc.

Definition at line 828 of file GenEvent.h.

Referenced by HepMC::convert_units(), event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), HepMC::detail::read_units(), and writePythiaStreamIO().

8.8.3.55 bool HepMC::GenEvent::valid_beam_particles () const

test to see if we have two valid beam particles

Examples:

testMass.cc.in.

Definition at line 570 of file GenEvent.cc.

References p, particles_begin(), and particles_end().

8.8.3.56 vertex_iterator HepMC::GenEvent::vertices_begin () [inline]

begin vertex iteration

Definition at line 416 of file GenEvent.h.

8.8.3.57 vertex_const_iterator HepMC::GenEvent::vertices_begin () const [inline]

begin vertex iteration

Examples:

example_UsingIterators.cc, and testHepMCIteration.cc.in.

Definition at line 353 of file GenEvent.h.

Referenced by HepMC::compareVertices(), GenEvent(), main(), swap(), write(), and HepMC::IO_HEPEVT::write_event().

8.8.3.58 bool HepMC::GenEvent::vertices_empty () const [inline]

return true if there are no vertex barcodes

Definition at line 811 of file GenEvent.h.

Referenced by delete_all_vertices(), and is_valid().

8.8.3.59 vertex_iterator HepMC::GenEvent::vertices_end () [inline]

end vertex iteration

Definition at line 420 of file GenEvent.h.

8.8.3.60 vertex_const_iterator HepMC::GenEvent::vertices_end () const [inline]

end vertex iteration

Examples:

example_UsingIterators.cc, and testHepMCIteration.cc.in.

Definition at line 357 of file GenEvent.h.

Referenced by HepMC::compareVertices(), GenEvent(), main(), print(), swap(), write(), and HepMC::IO_HEPEVT::write_event().

8.8.3.61 int HepMC::GenEvent::vertices_size () const [inline]

how many vertex barcodes exist?

Examples:

testMultipleCopies.cc.in.

Definition at line 808 of file GenEvent.h.

Referenced by HepMC::compareVertices(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.62 const WeightContainer & HepMC::GenEvent::weights () const [inline]

direct access to **WeightContainer** (p. 244)

Definition at line 673 of file GenEvent.h.

8.8.3.63 WeightContainer & HepMC::GenEvent::weights () [inline]

direct access to **WeightContainer** (p. 244)

direct access to the weights container is allowed. Thus you can use myevt.weights()[2]; to access element 2 of the weights. or use myevt.weights().push_back(mywgt); to add an element. and you can set the weights with myevt.weights() = myvector;

Definition at line 671 of file GenEvent.h.

Referenced by HepMC::compareWeights(), GenEvent(), print(), read(), write(), and HepMC::IO_AsciiParticles::write_event().

8.8.3.64 std::ostream & HepMC::GenEvent::write (std::ostream &)

Examples:

example_PythiaStreamIO.cc, and testStreamIO.cc.in.

Definition at line 72 of file GenEventStreamIO.cc.

References alphaQCD(), alphaQED(), beam_particles(), HepMC::WeightContainer::end(), event_number(), event_scale(), HepMC::StreamInfo::finished_first_event(), HepMC::get_stream_info(), heavy_ion(), length_unit(), momentum_unit(), mpi(), HepMC::Units::name(), HepMC::detail::output(), pdf_info(), HepMC::StreamInfo::set_finished_first_event(), signal_process_id(), signal_process_vertex(), HepMC::WeightContainer::size(), v, vertices_begin(), vertices_end(), vertices_size(), weights(), and HepMC::GenCrossSection::write().

Referenced by HepMC::operator<<(), and readPythiaStreamIO().

8.8.3.65 void HepMC::GenEvent::write_cross_section (std::ostream & ostr = std::cout) const

If the cross section is defined, write the cross section information to an output stream. If the output stream is not defined, use std::cout.

Examples:

testHepMC.cc.in.

Definition at line 607 of file GenEvent.cc.

References HepMC::GenCrossSection::cross_section(), cross_section(), and HepMC::GenCrossSection::cross_section_error().

Referenced by print().

8.8.3.66 void HepMC::GenEvent::write_units (std::ostream & os = std::cout) const

Write the unit information to an output stream. If the output stream is not defined, use std::cout.

Examples:

testHepMC.cc.in, and testStreamIO.cc.in.

Definition at line 601 of file GenEvent.cc.

References length_unit(), momentum_unit(), and HepMC::Units::name().

Referenced by print().

8.8.4 Friends And Related Function Documentation**8.8.4.1 friend class GenParticle [friend]**

Definition at line 151 of file GenEvent.h.

Referenced by GenEvent().

8.8.4.2 friend class GenVertex [friend]

Definition at line 152 of file GenEvent.h.

Referenced by GenEvent(), and read().

8.8.4.3 friend class particle_const_iterator [friend]

Definition at line 481 of file GenEvent.h.

Referenced by HepMC::GenEvent::particle_iterator::operator particle_const_iterator().

8.8.4.4 friend class particle_iterator [friend]

Definition at line 539 of file GenEvent.h.

8.8.4.5 friend class vertex_const_iterator [friend]

Definition at line 351 of file GenEvent.h.

Referenced by HepMC::GenEvent::vertex_iterator::operator vertex_const_iterator().

8.8.4.6 friend class vertex_iterator [friend]

Definition at line 414 of file GenEvent.h.

The documentation for this class was generated from the following files:

- **GenEvent.h**
- **GenEvent.cc**
- **GenEventStreamIO.cc**

8.9 HepMC::GenEvent::particle_const_iterator Class Reference

const particle iterator

```
#include <GenEvent.h>
```

Public Member Functions

- **particle_const_iterator** (const std::map< int, HepMC::GenParticle * >::const_iterator &i)
iterate over particles
- **particle_const_iterator** ()
- **particle_const_iterator** (const particle_const_iterator &i)
copy constructor
- **virtual ~particle_const_iterator** ()
- **particle_const_iterator & operator=** (const particle_const_iterator &i)
make a copy
- **GenParticle * operator *** (void) const
return a pointer to GenParticle (p. 99)
- **particle_const_iterator & operator++** (void)
Pre-fix increment.
- **particle_const_iterator operator++** (int)
Post-fix increment.
- **bool operator==** (const particle_const_iterator &a) const
equality
- **bool operator!=** (const particle_const_iterator &a) const
inequality

Protected Attributes

- std::map< int, HepMC::GenParticle * >::const_iterator m_map_iterator
const iterator to the GenParticle (p. 99) map

8.9.1 Detailed Description

const particle iterator

HepMC::GenEvent::particle_const_iterator (p. 86) is used to iterate over all particles in the event.

Examples:

`example_BuildEventFromScratch.cc`, `example_EventSelection.cc`, `example_MyPythia.cc`, `test-Mass.cc.in`, and `testMultipleCopies.cc.in`.

Definition at line 440 of file GenEvent.h.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 HepMC::GenEvent::particle_const_iterator::particle_const_iterator (const std::map< int, HepMC::GenParticle * >::const_iterator & i) [inline]

iterate over particles

Definition at line 445 of file GenEvent.h.

8.9.2.2 HepMC::GenEvent::particle_const_iterator::particle_const_iterator () [inline]

Definition at line 448 of file GenEvent.h.

8.9.2.3 HepMC::GenEvent::particle_const_iterator::particle_const_iterator (const particle_const_iterator & i) [inline]

copy constructor

Definition at line 450 of file GenEvent.h.

8.9.2.4 virtual HepMC::GenEvent::particle_const_iterator::~particle_const_iterator () [inline, virtual]

Definition at line 452 of file GenEvent.h.

8.9.3 Member Function Documentation

8.9.3.1 GenParticle* HepMC::GenEvent::particle_const_iterator::operator * (void) const [inline]

return a pointer to **GenParticle** (p. 99)

Definition at line 458 of file GenEvent.h.

References m_map_iterator.

8.9.3.2 bool HepMC::GenEvent::particle_const_iterator::operator!= (const particle_const_iterator & a) const [inline]

inequality

Definition at line 470 of file GenEvent.h.

References m_map_iterator.

8.9.3.3 particle_const_iterator HepMC::GenEvent::particle_const_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 464 of file GenEvent.h.

8.9.3.4 `particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator++ (void)`
[inline]

Pre-fix increment.

Definition at line 461 of file GenEvent.h.

References `m_map_iterator`.

8.9.3.5 `particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator= (const particle_const_iterator & i)` [inline]

make a copy

Definition at line 454 of file GenEvent.h.

References `m_map_iterator`.

8.9.3.6 `bool HepMC::GenEvent::particle_const_iterator::operator== (const particle_const_iterator & a) const` [inline]

equality

Definition at line 467 of file GenEvent.h.

References `m_map_iterator`.

8.9.4 Member Data Documentation

8.9.4.1 `std::map<int,HepMC::GenParticle*>::const_iterator HepMC::GenEvent::particle_const_iterator::m_map_iterator` [protected]

const iterator to the **GenParticle** (p. 99) map

Definition at line 474 of file GenEvent.h.

Referenced by `operator*()`, `operator!=()`, `operator++()`, `operator=()`, and `operator==()`.

The documentation for this class was generated from the following file:

- **GenEvent.h**

8.10 HepMC::GenEvent::particle_iterator Class Reference

non-const particle iterator

```
#include <GenEvent.h>
```

Public Member Functions

- **particle_iterator (const std::map< int, HepMC::GenParticle * >::iterator &i)**
iterate over particles
- **particle_iterator ()**
- **particle_iterator (const particle_iterator &i)**
copy constructor
- **virtual ~particle_iterator ()**
- **particle_iterator & operator= (const particle_iterator &i)**
make a copy
- **operator particle_const_iterator () const**
const particle iterator
- **GenParticle * operator * (void) const**
return pointer to GenParticle (p. 99)
- **particle_iterator & operator++ (void)**
Pre-fix increment.
- **particle_iterator operator++ (int)**
Post-fix increment.
- **bool operator== (const particle_iterator &a) const**
equality
- **bool operator!= (const particle_iterator &a) const**
inequality

Protected Attributes

- **std::map< int, HepMC::GenParticle * >::iterator m_map_iterator**
iterator for GenParticle (p. 99) map

8.10.1 Detailed Description

non-const particle iterator

HepMC::GenEvent::particle_iterator (p. 89) is used to iterate over all particles in the event.

Examples:

example_UsingIterators.cc, and testHepMCIteration.cc.in.

Definition at line 496 of file GenEvent.h.

8.10.2 Constructor & Destructor Documentation**8.10.2.1 HepMC::GenEvent::particle_iterator::particle_iterator (const std::map< int, HepMC::GenParticle * >::iterator & i) [inline]**

iterate over particles

Definition at line 501 of file GenEvent.h.

8.10.2.2 HepMC::GenEvent::particle_iterator::particle_iterator () [inline]

Definition at line 503 of file GenEvent.h.

8.10.2.3 HepMC::GenEvent::particle_iterator::particle_iterator (const particle_iterator & i) [inline]

copy constructor

Definition at line 505 of file GenEvent.h.

8.10.2.4 virtual HepMC::GenEvent::particle_iterator::~~particle_iterator () [inline, virtual]

Definition at line 506 of file GenEvent.h.

8.10.3 Member Function Documentation**8.10.3.1 GenParticle* HepMC::GenEvent::particle_iterator::operator * (void) const [inline]**

return pointer to **GenParticle** (p. 99)

Definition at line 516 of file GenEvent.h.

References m_map_iterator.

8.10.3.2 HepMC::GenEvent::particle_iterator::operator particle_const_iterator () const [inline]

const particle iterator

Definition at line 513 of file GenEvent.h.

References m_map_iterator, and HepMC::GenEvent::particle_const_iterator.

8.10.3.3 `bool HepMC::GenEvent::particle_iterator::operator!=(const particle_iterator & a) const` `[inline]`

inequality

Definition at line 528 of file GenEvent.h.

References `m_map_iterator`.

8.10.3.4 `particle_iterator HepMC::GenEvent::particle_iterator::operator++(int)` `[inline]`

Post-fix increment.

Definition at line 522 of file GenEvent.h.

8.10.3.5 `particle_iterator& HepMC::GenEvent::particle_iterator::operator++(void)` `[inline]`

Pre-fix increment.

Definition at line 519 of file GenEvent.h.

References `m_map_iterator`.

8.10.3.6 `particle_iterator& HepMC::GenEvent::particle_iterator::operator=(const particle_iterator & i)` `[inline]`

make a copy

Definition at line 508 of file GenEvent.h.

References `m_map_iterator`.

8.10.3.7 `bool HepMC::GenEvent::particle_iterator::operator==(const particle_iterator & a) const` `[inline]`

equality

Definition at line 525 of file GenEvent.h.

References `m_map_iterator`.

8.10.4 Member Data Documentation

8.10.4.1 `std::map<int,HepMC::GenParticle*>::iterator HepMC::GenEvent::particle_iterator::m_map_iterator` `[protected]`

iterator for `GenParticle` (p. 99) `map`

Definition at line 532 of file GenEvent.h.

Referenced by `operator*()`, `operator particle_const_iterator()`, `operator!=()`, `operator++()`, `operator=()`, and `operator==()`.

The documentation for this class was generated from the following file:

- `GenEvent.h`

8.11 HepMC::GenEvent::vertex_const_iterator Class Reference

const vertex iterator

```
#include <GenEvent.h>
```

Public Member Functions

- **vertex_const_iterator** (const std::map< int, HepMC::GenVertex *, std::greater< int > >::const_iterator &i)
constructor requiring vertex information
- **vertex_const_iterator** ()
- **vertex_const_iterator** (const vertex_const_iterator &i)
copy constructor
- **virtual ~vertex_const_iterator** ()
- **vertex_const_iterator & operator=** (const vertex_const_iterator &i)
make a copy
- **GenVertex * operator *** (void) const
return a pointer to a GenVertex (p. 109)
- **vertex_const_iterator & operator++** (void)
Pre-fix increment.
- **vertex_const_iterator operator++** (int)
Post-fix increment.
- **bool operator==** (const vertex_const_iterator &a) const
equality
- **bool operator!=** (const vertex_const_iterator &a) const
inequality

Protected Attributes

- std::map< int, HepMC::GenVertex *, std::greater< int > >::const_iterator m_map_iterator
const iterator to a vertex map

8.11.1 Detailed Description

const vertex iterator

HepMC::GenEvent::vertex_const_iterator (p. 92) is used to iterate over all vertices in the event.

Definition at line 310 of file GenEvent.h.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator (const std::map< int, HepMC::GenVertex *, std::greater< int > >::const_iterator & i) [inline]

constructor requiring vertex information

Definition at line 315 of file GenEvent.h.

8.11.2.2 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator () [inline]

Definition at line 319 of file GenEvent.h.

8.11.2.3 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator (const vertex_const_iterator & i) [inline]

copy constructor

Definition at line 321 of file GenEvent.h.

8.11.2.4 virtual HepMC::GenEvent::vertex_const_iterator::~~vertex_const_iterator () [inline, virtual]

Definition at line 323 of file GenEvent.h.

8.11.3 Member Function Documentation

8.11.3.1 GenVertex* HepMC::GenEvent::vertex_const_iterator::operator * (void) const [inline]

return a pointer to a **GenVertex** (p. 109)

Definition at line 328 of file GenEvent.h.

References m_map_iterator.

8.11.3.2 bool HepMC::GenEvent::vertex_const_iterator::operator!= (const vertex_const_iterator & a) const [inline]

inequality

Definition at line 339 of file GenEvent.h.

References m_map_iterator.

8.11.3.3 vertex_const_iterator HepMC::GenEvent::vertex_const_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 333 of file GenEvent.h.

8.11.3.4 `vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator++ (void)` [inline]

Pre-fix increment.

Definition at line 330 of file GenEvent.h.

References `m_map_iterator`.

8.11.3.5 `vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator= (const vertex_const_iterator & i)` [inline]

make a copy

Definition at line 325 of file GenEvent.h.

References `m_map_iterator`.

8.11.3.6 `bool HepMC::GenEvent::vertex_const_iterator::operator== (const vertex_const_iterator & a) const` [inline]

equality

Definition at line 336 of file GenEvent.h.

References `m_map_iterator`.

8.11.4 Member Data Documentation

8.11.4.1 `std::map<int,HepMC::GenVertex*,std::greater<int> >::const_iterator HepMC::GenEvent::vertex_const_iterator::m_map_iterator` [protected]

const iterator to a vertex map

Definition at line 344 of file GenEvent.h.

Referenced by `operator*()`, `operator!=()`, `operator++()`, `operator=()`, and `operator==()`.

The documentation for this class was generated from the following file:

- **GenEvent.h**

8.12 HepMC::GenEvent::vertex_iterator Class Reference

non-const vertex iterator

```
#include <GenEvent.h>
```

Public Member Functions

- **vertex_iterator (const std::map< int, HepMC::GenVertex *, std::greater< int > >::iterator &i)**

constructor requiring vertex information

- **vertex_iterator ()**
- **vertex_iterator (const vertex_iterator &i)**

copy constructor

- **virtual ~vertex_iterator ()**
- **vertex_iterator & operator= (const vertex_iterator &i)**

make a copy

- **operator vertex_const_iterator () const**

const vertex iterator

- **GenVertex * operator * (void) const**

return a pointer to a GenVertex (p. 109)

- **vertex_iterator & operator++ (void)**

Pre-fix increment.

- **vertex_iterator operator++ (int)**

Post-fix increment.

- **bool operator== (const vertex_iterator &a) const**

equality

- **bool operator!= (const vertex_iterator &a) const**

inequality

Protected Attributes

- **std::map< int, HepMC::GenVertex *, std::greater< int > >::iterator m_map_iterator**

iterator to the vertex map

8.12.1 Detailed Description

non-const vertex iterator

HepMC::GenEvent::vertex_iterator (p. 95) is used to iterate over all vertices in the event.

Examples:

`example_UsingIterators.cc`, and `testHepMCIteration.cc.in`.

Definition at line 367 of file `GenEvent.h`.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 HepMC::GenEvent::vertex_iterator::vertex_iterator (const std::map< int, HepMC::GenVertex *, std::greater< int > >::iterator & i) [inline]

constructor requiring vertex information

Definition at line 372 of file `GenEvent.h`.

8.12.2.2 HepMC::GenEvent::vertex_iterator::vertex_iterator () [inline]

Definition at line 376 of file `GenEvent.h`.

8.12.2.3 HepMC::GenEvent::vertex_iterator::vertex_iterator (const vertex_iterator & i) [inline]

copy constructor

Definition at line 378 of file `GenEvent.h`.

8.12.2.4 virtual HepMC::GenEvent::vertex_iterator::~~vertex_iterator () [inline, virtual]

Definition at line 379 of file `GenEvent.h`.

8.12.3 Member Function Documentation

8.12.3.1 GenVertex* HepMC::GenEvent::vertex_iterator::operator * (void) const [inline]

return a pointer to a **GenVertex** (p. 109)

Definition at line 389 of file `GenEvent.h`.

References `m_map_iterator`.

8.12.3.2 HepMC::GenEvent::vertex_iterator::operator vertex_const_iterator () const [inline]

const vertex iterator

Definition at line 386 of file GenEvent.h.

References `m_map_iterator`, and `HepMC::GenEvent::vertex_const_iterator`.

8.12.3.3 `bool HepMC::GenEvent::vertex_iterator::operator!=(const vertex_iterator & a) const`
[inline]

inequality

Definition at line 401 of file GenEvent.h.

References `m_map_iterator`.

8.12.3.4 `vertex_iterator HepMC::GenEvent::vertex_iterator::operator++(int)` [inline]

Post-fix increment.

Definition at line 395 of file GenEvent.h.

8.12.3.5 `vertex_iterator& HepMC::GenEvent::vertex_iterator::operator++(void)` [inline]

Pre-fix increment.

Definition at line 392 of file GenEvent.h.

References `m_map_iterator`.

8.12.3.6 `vertex_iterator& HepMC::GenEvent::vertex_iterator::operator=(const vertex_iterator & i)` [inline]

make a copy

Definition at line 381 of file GenEvent.h.

References `m_map_iterator`.

8.12.3.7 `bool HepMC::GenEvent::vertex_iterator::operator==(const vertex_iterator & a) const`
[inline]

equality

Definition at line 398 of file GenEvent.h.

References `m_map_iterator`.

8.12.4 Member Data Documentation

8.12.4.1 `std::map<int,HepMC::GenVertex*,std::greater<int>>::iterator`
`HepMC::GenEvent::vertex_iterator::m_map_iterator` [protected]

iterator to the vertex map

Definition at line 406 of file GenEvent.h.

Referenced by `operator*()`, `operator vertex_const_iterator()`, `operator!=()`, `operator++()`, `operator=()`, and `operator==()`.

The documentation for this class was generated from the following file:

- **GenEvent.h**

8.13 HepMC::GenParticle Class Reference

The **GenParticle** (p. 99) class contains information about generated particles.

```
#include <GenParticle.h>
```

Public Member Functions

- **GenParticle (void)**
default constructor
- **GenParticle (const FourVector &momentum, int pdg_id, int status=0, const Flow &its-flow=Flow(), const Polarization &polar=Polarization(0, 0))**
constructor requires momentum and particle ID
- **GenParticle (const GenParticle &inparticle)**
shallow copy.
- **virtual ~GenParticle ()**
- **void swap (GenParticle &other)**
swap
- **GenParticle & operator= (const GenParticle &inparticle)**
- **bool operator== (const GenParticle &) const**
check for equality
- **bool operator!= (const GenParticle &) const**
check for inequality
- **void print (std::ostream &ostr=std::cout) const**
dump this particle's full info to ostr
- **operator HepMC::FourVector () const**
conversion operator
- **const FourVector & momentum () const**
standard 4 momentum
- **int pdg_id () const**
particle ID
- **int status () const**
HEPEVT decay status.
- **const Flow & flow () const**
particle flow
- **int flow (int code_index) const**
particle flow index

- **const Polarization & polarization () const**
polarization information
- **GenVertex * production_vertex () const**
pointer to the production vertex
- **GenVertex * end_vertex () const**
pointer to the decay vertex
- **GenEvent * parent_event () const**
pointer to the event that owns this particle
- **double generated_mass () const**
mass as generated
- **double generatedMass () const**
generatedMass() (p. 103) is included for backwards compatibility with CLHEP (p. 17) HepMC (p. 19)
- **int barcode () const**
particle barcode
- **bool is_undecayed () const**
Convenience method. Returns true if status==1.
- **bool has_decayed () const**
Convenience method. Returns true if status==2.
- **bool is_beam () const**
- **bool suggest_barcode (int the_bar_code)**
In general there is no reason to "suggest_barcode".
- **void set_momentum (const FourVector &vec4)**
set standard 4 momentum
- **void set_pdg_id (int id)**
set particle ID
- **void set_status (int status=0)**
set decay status
- **void set_flow (const Flow &f)**
set particle flow
- **void set_flow (int code_index, int code=0)**
- **void set_polarization (const Polarization &pol=Polarization(0, 0))**
set polarization
- **void set_generated_mass (const double &m)**
define the actual generated mass

- **void setGeneratedMass (const double &m)**
setGeneratedMass() (p. 107) is included for backwards compatibility with CLHEP (p. 17) HepMC (p. 19)

Protected Member Functions

- **void set_production_vertex_ (GenVertex *productionvertex=0)**
set production vertex - for internal use only
- **void set_end_vertex_ (GenVertex *decayvertex=0)**
set decay vertex - for internal use only
- **void set_barcode_ (int the_bar_code)**
for use by GenEvent (p. 64) only
- **void convert_momentum (const double &)**

Friends

- class **GenVertex**
- class **GenEvent**
- **std::ostream & operator<< (std::ostream &, const GenParticle &)**
print particle

8.13.1 Detailed Description

The **GenParticle** (p. 99) class contains information about generated particles.

HepMC::GenParticle (p. 99) contains momentum, generated mass, particle ID, decay status, flow, polarization, pointers to production and decay vertices and a unique barcode identifier.

Examples:

example_BuildEventFromScratch.cc, example_UsingIterators.cc, testFlow.cc, testMass.cc.in, and testPrintBug.cc.

Definition at line 55 of file GenParticle.h.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 HepMC::GenParticle::GenParticle (void)

default constructor

Definition at line 14 of file GenParticle.cc.

8.13.2.2 **HepMC::GenParticle::GenParticle** (const FourVector & *momentum*, int *pdg_id*, int *status* = 0, const Flow & *itsflow* = Flow(), const Polarization & *polar* = Polarization(0, 0))

constructor requires momentum and particle ID

Definition at line 23 of file GenParticle.cc.

References set_flow().

8.13.2.3 **HepMC::GenParticle::GenParticle** (const GenParticle & *inparticle*)

shallow copy.

Shallow copy: does not copy the vertex pointers (note - impossible to copy vertex pointers which having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 37 of file GenParticle.cc.

References barcode(), set_end_vertex_(), set_production_vertex_(), and suggest_barcode().

8.13.2.4 **HepMC::GenParticle::~~GenParticle** () [virtual]

Definition at line 58 of file GenParticle.cc.

References parent_event(), and HepMC::GenEvent::remove_barcode().

8.13.3 Member Function Documentation

8.13.3.1 **int HepMC::GenParticle::barcode** () const [inline]

particle barcode

The barcode is the particle's reference number, every vertex in the event has a unique barcode. Particle barcodes are positive numbers, vertex barcodes are negative numbers.

Please note that the barcodes are intended for internal use within **HepMC** (p. 19) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Examples:

testFlow.cc.

Definition at line 238 of file GenParticle.h.

Referenced by GenParticle(), main(), HepMC::operator<<(), print(), set_end_vertex_(), and set_production_vertex_().

8.13.3.2 **void HepMC::GenParticle::convert_momentum** (const double &) [protected]

scale the momentum vector and generated mass this method is only for use by **GenEvent** (p. 64)

Definition at line 246 of file GenParticle.cc.

References HepMC::FourVector::e(), HepMC::FourVector::px(), HepMC::FourVector::py(), and HepMC::FourVector::pz().

8.13.3.3 GenVertex * HepMC::GenParticle::end_vertex () const [inline]

pointer to the decay vertex

Definition at line 207 of file GenParticle.h.

Referenced by HepMC::GenVertex::add_particle_in(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::operator<<(), parent_event(), print(), and HepMC::GenVertex::remove_particle().

8.13.3.4 int HepMC::GenParticle::flow (int *code_index*) const [inline]

particle flow index

Definition at line 211 of file GenParticle.h.

References HepMC::Flow::icode().

8.13.3.5 const Flow & HepMC::GenParticle::flow () const [inline]

particle flow

Examples:

testFlow.cc.

Definition at line 209 of file GenParticle.h.

Referenced by main().

8.13.3.6 double HepMC::GenParticle::generated_mass () const

mass as generated

Because of precision issues, the generated mass is not always the same as the mass calculated from the momentum 4 vector. If the generated mass has been set, then **generated_mass()** (p. 103) returns that value. If the generated mass has not been set, then **generated_mass()** (p. 103) returns the mass calculated from the momentum 4 vector.

Definition at line 236 of file GenParticle.cc.

Referenced by generatedMass(), and operator==().

8.13.3.7 double HepMC::GenParticle::generatedMass () const [inline]

generatedMass() (p. 103) is included for backwards compatibility with **CLHEP** (p. 17) **HepMC** (p. 19)

Definition at line 116 of file GenParticle.h.

References generated_mass().

8.13.3.8 bool HepMC::GenParticle::has_decayed () const [inline]

Convenience method. Returns true if status==2.

Definition at line 245 of file GenParticle.h.

8.13.3.9 bool HepMC::GenParticle::is_beam () const [inline]

Convenience method. Returns true if status==4 Note that using status 4 for beam particles is a new convention which may not have been implemented by the code originating this **GenEvent** (p. 64).

Definition at line 248 of file GenParticle.h.

8.13.3.10 bool HepMC::GenParticle::is_undecayed () const [inline]

Convenience method. Returns true if status==1.

Definition at line 242 of file GenParticle.h.

8.13.3.11 const FourVector & HepMC::GenParticle::momentum () const [inline]

standard 4 momentum

Definition at line 197 of file GenParticle.h.

Referenced by HepMC::operator<<(), operator==(), and print().

8.13.3.12 HepMC::GenParticle::operator HepMC::FourVector () const [inline]

conversion operator

Definition at line 194 of file GenParticle.h.

8.13.3.13 bool HepMC::GenParticle::operator!= (const GenParticle &) const

check for inequality

Definition at line 102 of file GenParticle.cc.

8.13.3.14 GenParticle & HepMC::GenParticle::operator= (const GenParticle & *inparticle*)

shallow.

Shallow: does not copy the vertex pointers (note - impossible to copy vertex pointers which having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 77 of file GenParticle.cc.

References swap().

8.13.3.15 bool HepMC::GenParticle::operator== (const GenParticle &) const

check for equality

consistent with the definition of the copy constructor as a shallow constructor,.. this operator does not test the vertex pointers. Does not compare barcodes.

Definition at line 89 of file GenParticle.cc.

References generated_mass(), m_flow, momentum(), pdg_id(), polarization(), and status().

8.13.3.16 GenEvent * HepMC::GenParticle::parent_event () const

pointer to the event that owns this particle

Definition at line 123 of file GenParticle.cc.

References end_vertex(), HepMC::GenVertex::parent_event(), and production_vertex().

Referenced by set_end_vertex_(), set_production_vertex_(), suggest_barcode(), and ~GenParticle().

8.13.3.17 int HepMC::GenParticle::pdg_id () const [inline]

particle ID

Definition at line 200 of file GenParticle.h.

Referenced by HepMC::operator<<(), operator==(), and print().

8.13.3.18 const Polarization & HepMC::GenParticle::polarization () const [inline]

polarization information

Definition at line 214 of file GenParticle.h.

Referenced by operator==(), and print().

8.13.3.19 void HepMC::GenParticle::print (std::ostream & ostr = std::cout) const

dump this particle's full info to ostr

Dump this particle's full info to ostr, where by default particle.print(); will dump to cout.

Definition at line 106 of file GenParticle.cc.

References HepMC::GenVertex::barcode(), barcode(), HepMC::FourVector::e(), end_vertex(), momentum(), pdg_id(), polarization(), production_vertex(), HepMC::FourVector::px(), HepMC::FourVector::py(), HepMC::FourVector::pz(), and status().

8.13.3.20 GenVertex * HepMC::GenParticle::production_vertex () const [inline]

pointer to the production vertex

Definition at line 204 of file GenParticle.h.

Referenced by HepMC::GenVertex::add_particle_out(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), parent_event(), print(), and HepMC::GenVertex::remove_particle().

8.13.3.21 void HepMC::GenParticle::set_barcode_ (int *the_bar_code*) [inline, protected]

for use by **GenEvent** (p. 64) only

Definition at line 240 of file GenParticle.h.

Referenced by suggest_barcode().

8.13.3.22 void HepMC::GenParticle::set_end_vertex_ (GenVertex * *decayvertex* = 0) [protected]

set decay vertex - for internal use only

Definition at line 142 of file GenParticle.cc.

References barcode(), parent_event(), and HepMC::GenEvent::remove_barcode().

Referenced by HepMC::GenVertex::add_particle_in(), GenParticle(), and HepMC::GenVertex::remove_particle().

8.13.3.23 void HepMC::GenParticle::set_flow (int *code_index*, int *code* = 0) [inline]

set particle flow index

Definition at line 226 of file GenParticle.h.

References HepMC::Flow::set_icode(), and HepMC::Flow::set_unique_icode().

8.13.3.24 void HepMC::GenParticle::set_flow (const Flow & *f*) [inline]

set particle flow

Examples:

testFlow.cc.

Definition at line 224 of file GenParticle.h.

Referenced by GenParticle(), and main().

8.13.3.25 void HepMC::GenParticle::set_generated_mass (const double & *m*)

define the actual generated mass

If you do not call **set_generated_mass()** (p. 106), then **generated_mass()** (p. 103) will simply return the mass calculated from **momentum()** (p. 104)

Definition at line 240 of file GenParticle.cc.

Referenced by setGeneratedMass().

8.13.3.26 void HepMC::GenParticle::set_momentum (const FourVector & *vec4*) [inline]

set standard 4 momentum

Definition at line 217 of file GenParticle.h.

8.13.3.27 void HepMC::GenParticle::set_pdg_id (int *id*) [inline]

set particle ID

Definition at line 220 of file GenParticle.h.

8.13.3.28 void HepMC::GenParticle::set_polarization (const Polarization & pol = Polarization(0, 0)) [inline]

set polarization

Definition at line 235 of file GenParticle.h.

8.13.3.29 void HepMC::GenParticle::set_production_vertex_ (GenVertex * productionvertex = 0) [protected]

set production vertex - for internal use only

Definition at line 129 of file GenParticle.cc.

References barcode(), parent_event(), and HepMC::GenEvent::remove_barcode().

Referenced by HepMC::GenVertex::add_particle_out(), GenParticle(), and HepMC::GenVertex::remove_particle().

8.13.3.30 void HepMC::GenParticle::set_status (int status = 0) [inline]

set decay status

Definition at line 222 of file GenParticle.h.

8.13.3.31 void HepMC::GenParticle::setGeneratedMass (const double & m) [inline]

setGeneratedMass() (p. 107) is included for backwards compatibility with CLHEP (p. 17) HepMC (p. 19)

Definition at line 159 of file GenParticle.h.

References set_generated_mass().

8.13.3.32 int HepMC::GenParticle::status () const [inline]

HEPEVT decay status.

Definition at line 202 of file GenParticle.h.

Referenced by HepMC::operator<<(), operator==(), and print().

8.13.3.33 bool HepMC::GenParticle::suggest_barcode (int the_bar_code)

In general there is no reason to "suggest_barcode".

allows a barcode to be suggested for this particle. In general it is better to let the event pick the barcode for you, which is automatic. Returns TRUE if the suggested barcode has been accepted (i.e. the suggested barcode has not already been used in the event, and so it was used). Returns FALSE if the suggested barcode was rejected, or if the particle is not yet part of an event, such that it is not yet possible to know if the suggested barcode will be accepted).

Definition at line 153 of file GenParticle.cc.

References parent_event(), HepMC::GenEvent::set_barcode(), and set_barcode_().

Referenced by GenParticle().

8.13.3.34 void HepMC::GenParticle::swap (GenParticle & *other*)

swap

Definition at line 63 of file GenParticle.cc.

References m_barcode, m_end_vertex, m_flow, m_generated_mass, m_momentum, m_pdg_id, m_polarization, m_production_vertex, m_status, HepMC::Polarization::swap(), HepMC::Flow::swap(), and HepMC::FourVector::swap().

Referenced by operator=().

8.13.4 Friends And Related Function Documentation**8.13.4.1 friend class GenEvent [friend]**

Definition at line 58 of file GenParticle.h.

8.13.4.2 friend class GenVertex [friend]

Definition at line 57 of file GenParticle.h.

8.13.4.3 std::ostream& operator<< (std::ostream & *ostr*, const GenParticle & *part*) [friend]

print particle

Definition at line 189 of file GenParticle.cc.

The documentation for this class was generated from the following files:

- **GenParticle.h**
- **GenParticle.cc**

8.14 HepMC::GenVertex Class Reference

GenVertex (p. 109) contains information about decay vertices.

```
#include <GenVertex.h>
```

Public Types

- `typedef std::vector< HepMC::GenParticle * >::const_iterator particles_in_const_iterator`
const iterator for incoming particles
- `typedef std::vector< HepMC::GenParticle * >::const_iterator particles_out_const_iterator`
const iterator for outgoing particles

Public Member Functions

- `GenVertex (const FourVector &position=FourVector(0, 0, 0, 0), int id=0, const Weight-Container &weights=std::vector< double >())`
default constructor
- `GenVertex (const GenVertex &invertex)`
shallow copy
- `virtual ~GenVertex ()`
- `void swap (GenVertex &other)`
swap
- `GenVertex & operator= (const GenVertex &invertex)`
shallow
- `bool operator== (const GenVertex &a) const`
equality
- `bool operator!= (const GenVertex &a) const`
inequality
- `void print (std::ostream &ostr=std::cout) const`
print vertex information
- `double check_momentum_conservation () const`
|Sum (mom_in-mom_out)|
- `void add_particle_in (GenParticle *inparticle)`
add incoming particle
- `void add_particle_out (GenParticle *outparticle)`
add outgoing particle
- `GenParticle * remove_particle (GenParticle *particle)`

remove a particle

- **operator HepMC::FourVector () const**
conversion operator
- **operator HepMC::ThreeVector () const**
conversion operator
- **GenEvent * parent_event () const**
pointer to the event that owns this vertex
- **ThreeVector point3d () const**
vertex position
- **const FourVector & position () const**
vertex position and time
- **void set_position (const FourVector &position=FourVector(0, 0, 0, 0))**
set vertex position and time
- **int id () const**
vertex ID
- **void set_id (int id)**
set vertex ID
- **int barcode () const**
unique identifier
- **bool suggest_barcode (int the_bar_code)**
In general there is no reason to "suggest_barcode".
- **WeightContainer & weights ()**
direct access to the weights container is allowed.
- **const WeightContainer & weights () const**
const direct access to the weights container
- **particles_in_const_iterator particles_in_const_begin () const**
begin iteration of incoming particles
- **particles_in_const_iterator particles_in_const_end () const**
end iteration of incoming particles
- **particles_out_const_iterator particles_out_const_begin () const**
begin iteration of outgoing particles
- **particles_out_const_iterator particles_out_const_end () const**
end iteration of outgoing particles

- **int particles_in_size () const**
number of incoming particles
- **int particles_out_size () const**
number of outgoing particles
- **vertex_iterator vertices_begin (IteratorRange range=relatives)**
begin vertex range
- **vertex_iterator vertices_end (IteratorRange)**
end vertex range
- **particle_iterator particles_begin (IteratorRange range=relatives)**
begin particle range
- **particle_iterator particles_end (IteratorRange)**
end particle range

Protected Member Functions

- **void set_parent_event_ (GenEvent *evt)**
set parent event
- **void set_barcode_ (int the_bar_code)**
set identifier
- **void change_parent_event_ (GenEvent *evt)**
for use with swap
- **int edges_size (IteratorRange range=family) const**
size
- **edge_iterator edges_begin (IteratorRange range=family) const**
begin range
- **edge_iterator edges_end (IteratorRange) const**
end range
- **void delete_adopted_particles ()**
for internal use only
- **void remove_particle_in (GenParticle *)**
for internal use only - remove particle from incoming list
- **void remove_particle_out (GenParticle *)**
for internal use only - remove particle from outgoing list
- **void convert_position (const double &)**

Friends

- class **GenEvent**
- class **edge_iterator**
- class **vertex_iterator**
- class **particle_iterator**
- **std::ostream & operator<<** (**std::ostream &**, **const GenVertex &**)
print vertex information

Classes

- class **edge_iterator**
edge iterator
- class **particle_iterator**
particle iterator
- class **vertex_iterator**
vertex iterator

8.14.1 Detailed Description

GenVertex (p. 109) contains information about decay vertices.

HepMC::GenVertex (p. 109) contains the position in space and time of a decay. It also contains lists of incoming and outgoing particles.

Examples:

example_BuildEventFromScratch.cc, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 47 of file **GenVertex.h**.

8.14.2 Member Typedef Documentation

8.14.2.1 **typedef std::vector<HepMC::GenParticle*>::const_iterator HepMC::GenVertex::particles_in_const_iterator**

const iterator for incoming particles

Definition at line 136 of file **GenVertex.h**.

8.14.2.2 **typedef std::vector<HepMC::GenParticle*>::const_iterator HepMC::GenVertex::particles_out_const_iterator**

const iterator for outgoing particles

Definition at line 139 of file **GenVertex.h**.

8.14.3 Constructor & Destructor Documentation

8.14.3.1 HepMC::GenVertex::GenVertex (const FourVector & *position* = FourVector(0, 0, 0, 0), int *id* = 0, const WeightContainer & *weights* = std::vector< double >())

default constructor

Definition at line 14 of file GenVertex.cc.

8.14.3.2 HepMC::GenVertex::GenVertex (const GenVertex & *invertex*)

shallow copy

Shallow copy: does not copy the FULL list of particle pointers. Creates a copy of - invertex

- outgoing particles of invertex, but sets the decay vertex of these particles to NULL
- all incoming particles which do not have a creation vertex. (i.e. it creates copies of all particles which it owns) (note - impossible to copy the FULL list of particle pointers while having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 23 of file GenVertex.cc.

References add_particle_in(), add_particle_out(), barcode(), particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), particles_out_const_end(), and suggest_barcode().

8.14.3.3 HepMC::GenVertex::~~GenVertex () [virtual]

Definition at line 63 of file GenVertex.cc.

References delete_adopted_particles(), parent_event(), and HepMC::GenEvent::remove_barcode().

8.14.4 Member Function Documentation

8.14.4.1 void HepMC::GenVertex::add_particle_in (GenParticle * *inparticle*)

add incoming particle

Examples:

example_BuildEventFromScratch.cc, testFlow.cc, and testPrintBug.cc.

Definition at line 273 of file GenVertex.cc.

References HepMC::GenParticle::end_vertex(), remove_particle_in(), and HepMC::GenParticle::set_end_vertex().

Referenced by HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::fill_next_event(), GenVertex(), main(), and HepMC::GenEvent::read().

8.14.4.2 void HepMC::GenVertex::add_particle_out (GenParticle * *outparticle*)

add outgoing particle

Examples:

example_BuildEventFromScratch.cc, testFlow.cc, and testPrintBug.cc.

Definition at line 284 of file GenVertex.cc.

References HepMC::GenParticle::production_vertex(), remove_particle_out(), and HepMC::GenParticle::set_production_vertex().

Referenced by HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), GenVertex(), and main().

8.14.4.3 int HepMC::GenVertex::barcode () const [inline]

unique identifier

The barcode is the vertex's reference number, every vertex in the event has a unique barcode. Vertex barcodes are negative numbers, particle barcodes are positive numbers.

Please note that the barcodes are intended for internal use within **HepMC** (p. 19) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Definition at line 400 of file GenVertex.h.

Referenced by HepMC::GenEvent::add_vertex(), HepMC::compareVertex(), GenVertex(), HepMC::operator<(), print(), HepMC::GenParticle::print(), HepMC::GenEvent::print(), HepMC::GenEvent::remove_vertex(), set_parent_event_(), and HepMC::IO_AsciiParticles::write_event().

8.14.4.4 void HepMC::GenVertex::change_parent_event_ (GenEvent * *evt*) [protected]

for use with swap

Definition at line 419 of file GenVertex.cc.

8.14.4.5 double HepMC::GenVertex::check_momentum_conservation () const

|Sum (mom_in-mom_out)|

finds the difference between the total momentum out and the total momentum in vectors, and returns the magnitude of this vector i.e. returns | vec{p_in} - vec{p_out} |

Definition at line 253 of file GenVertex.cc.

References particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), and particles_out_const_end().

8.14.4.6 void HepMC::GenVertex::convert_position (const double &) [protected]

scale the position vector this method is only for use by **GenEvent** (p. 64)

Definition at line 918 of file GenVertex.cc.

References HepMC::FourVector::t(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

8.14.4.7 void HepMC::GenVertex::delete_adopted_particles() [protected]

for internal use only

deletes all particles which this vertex owns to be used by the vertex destructor and operator=

Definition at line 329 of file GenVertex.cc.

Referenced by ~GenVertex().

8.14.4.8 GenVertex::edge_iterator HepMC::GenVertex::edges_begin (IteratorRange range = family) const [inline, protected]

begin range

Definition at line 460 of file GenVertex.h.

Referenced by HepMC::GenVertex::vertex_iterator::vertex_iterator().

8.14.4.9 GenVertex::edge_iterator HepMC::GenVertex::edges_end (IteratorRange) const [inline, protected]

end range

Definition at line 465 of file GenVertex.h.

Referenced by HepMC::GenVertex::vertex_iterator::operator++(), and HepMC::GenVertex::vertex_iterator::vertex_iterator().

8.14.4.10 int HepMC::GenVertex::edges_size (IteratorRange range = family) const [protected]

size

Definition at line 595 of file GenVertex.cc.

References HepMC::children, HepMC::family, and HepMC::parents.

8.14.4.11 int HepMC::GenVertex::id () const [inline]

vertex ID

we don't define what you use the id for – but we imagine, for example it might code the meaning of the **weights()** (p. 121)

Definition at line 398 of file GenVertex.h.

Referenced by print().

8.14.4.12 HepMC::GenVertex::operator HepMC::FourVector () const [inline]

conversion operator

Definition at line 386 of file GenVertex.h.

References position().

8.14.4.13 HepMC::GenVertex::operator HepMC::ThreeVector () const [inline]

conversion operator

Definition at line 388 of file GenVertex.h.

References point3d().

8.14.4.14 bool HepMC::GenVertex::operator!= (const GenVertex & a) const

inequality

Definition at line 140 of file GenVertex.cc.

8.14.4.15 GenVertex & HepMC::GenVertex::operator= (const GenVertex & *invertex*)

shallow

Shallow: does not copy the FULL list of particle pointers. Creates a copy of - invertex

- outgoing particles of invertex, but sets the decay vertex of these particles to NULL
- all incoming particles which do not have a creation vertex.
- it does not alter *this's m_event (!) (i.e. it creates copies of all particles which it owns) (note - impossible to copy the FULL list of particle pointers while having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 82 of file GenVertex.cc.

References swap().

8.14.4.16 bool HepMC::GenVertex::operator== (const GenVertex & a) const

equality

Returns true if the positions and the particles in the lists of a and this are identical. Does not compare barcodes. Note that it is impossible for two vertices to point to the same particle's address, so we need to do more than just compare the particle pointers

Definition at line 103 of file GenVertex.cc.

References particles_in_const_begin(), particles_in_const_end(), particles_in_size(), particles_out_const_begin(), particles_out_const_end(), particles_out_size(), and position().

8.14.4.17 GenEvent * HepMC::GenVertex::parent_event () const [inline]

pointer to the event that owns this vertex

Definition at line 392 of file GenVertex.h.

Referenced by HepMC::GenEvent::add_vertex(), HepMC::GenParticle::parent_event(), HepMC::GenEvent::remove_vertex(), suggest_barcode(), and ~GenVertex().

8.14.4.18 GenVertex::particle_iterator HepMC::GenVertex::particles_begin (IteratorRange *range* = relatives) [inline]

begin particle range

Definition at line 509 of file GenVertex.h.

References particle_iterator.

Referenced by HepMC::Flow::connected_partners(), and HepMC::Flow::dangling_connected_partners().

8.14.4.19 GenVertex::particle_iterator HepMC::GenVertex::particles_end (IteratorRange) [inline]

end particle range

Definition at line 514 of file GenVertex.h.

References particle_iterator.

Referenced by HepMC::Flow::connected_partners(), and HepMC::Flow::dangling_connected_partners().

8.14.4.20 GenVertex::particles_in_const_iterator HepMC::GenVertex::particles_in_const_begin () const [inline]

begin iteration of incoming particles

Definition at line 419 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), GenVertex(), operator==(), print(), and set_parent_event_().

8.14.4.21 GenVertex::particles_in_const_iterator HepMC::GenVertex::particles_in_const_end () const [inline]

end iteration of incoming particles

Definition at line 424 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), GenVertex(), operator==(), print(), and set_parent_event_().

8.14.4.22 int HepMC::GenVertex::particles_in_size () const [inline]

number of incoming particles

Definition at line 438 of file GenVertex.h.

Referenced by HepMC::compareVertex(), and operator==().

8.14.4.23 **GenVertex::particles_out_const_iterator** HepMC::GenVertex::particles_out_const_begin () const [inline]

begin iteration of outgoing particles

Definition at line 429 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), GenVertex(), operator==(), print(), and set_parent_event_().

8.14.4.24 **GenVertex::particles_out_const_iterator** HepMC::GenVertex::particles_out_const_end () const [inline]

end iteration of outgoing particles

Definition at line 434 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), GenVertex(), operator==(), print(), and set_parent_event_().

8.14.4.25 **int** HepMC::GenVertex::particles_out_size () const [inline]

number of outgoing particles

Definition at line 442 of file GenVertex.h.

Referenced by HepMC::compareVertex(), and operator==().

8.14.4.26 **ThreeVector** HepMC::GenVertex::point3d () const [inline]

vertex position

Definition at line 394 of file GenVertex.h.

References HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

Referenced by operator HepMC::ThreeVector().

8.14.4.27 **const FourVector &** HepMC::GenVertex::position () const [inline]

vertex position and time

Definition at line 390 of file GenVertex.h.

Referenced by HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::compareVertex(), operator HepMC::FourVector(), HepMC::operator<<(), operator==(), print(), and set_position().

8.14.4.28 **void** HepMC::GenVertex::print (std::ostream & ostr = std::cout) const

print vertex information

Definition at line 145 of file GenVertex.cc.

References barcode(), HepMC::WeightContainer::end(), id(), particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), particles_out_const_end(), position(), HepMC::Weight-

Container::size(), HepMC::FourVector::t(), weights(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

Referenced by HepMC::IO_HERWIG::build_production_vertex().

8.14.4.29 GenParticle * HepMC::GenVertex::remove_particle (GenParticle * *particle*)

remove a particle

remove_particle finds *particle in the in and/or out list and removes it from these lists ... it DOES NOT DELETE THE PARTICLE or its relations. You could delete the particle too as follows: delete vtx->remove_particle(particle);

this finds *particle in the in and/or out list and removes it from these lists ... it DOES NOT DELETE THE PARTICLE or its relations. you could delete the particle too as follows: delete vtx->remove_particle(particle); or if the particle has an end vertex, you could: delete vtx->remove_particle(particle)->end_vertex(); which would delete the particle's end vertex, and thus would also delete the particle, since the particle would be owned by the end vertex.

Definition at line 295 of file GenVertex.cc.

References HepMC::GenParticle::end_vertex(), HepMC::GenParticle::production_vertex(), remove_particle_in(), remove_particle_out(), HepMC::GenParticle::set_end_vertex(), and HepMC::GenParticle::set_production_vertex().

8.14.4.30 void HepMC::GenVertex::remove_particle_in (GenParticle *) [protected]

for internal use only - remove particle from incoming list

this finds *particle in m_particles_in and removes it from that list

Definition at line 317 of file GenVertex.cc.

References HepMC::already_in_vector().

Referenced by add_particle_in(), and remove_particle().

8.14.4.31 void HepMC::GenVertex::remove_particle_out (GenParticle *) [protected]

for internal use only - remove particle from outgoing list

this finds *particle in m_particles_out and removes it from that list

Definition at line 323 of file GenVertex.cc.

References HepMC::already_in_vector().

Referenced by add_particle_out(), and remove_particle().

8.14.4.32 void HepMC::GenVertex::set_barcode_ (int *the_bar_code*) [inline, protected]

set identifier

Definition at line 401 of file GenVertex.h.

Referenced by suggest_barcode().

8.14.4.33 void HepMC::GenVertex::set_id (int *id*) [inline]

set vertex ID

Definition at line 412 of file GenVertex.h.

8.14.4.34 void HepMC::GenVertex::set_parent_event_ (GenEvent * *evt*) [protected]

set parent event

only the **GenEvent** (p. 64) (friend) is allowed to set the parent_event, and barcode. It is done automatically anytime you add a vertex to an event

Definition at line 388 of file GenVertex.cc.

References barcode(), particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), particles_out_const_end(), HepMC::GenEvent::remove_barcode(), and HepMC::GenEvent::set_barcode().

Referenced by HepMC::GenEvent::add_vertex(), and HepMC::GenEvent::remove_vertex().

8.14.4.35 void HepMC::GenVertex::set_position (const FourVector & *position* = FourVector(0, 0, 0, 0)) [inline]

set vertex position and time

Definition at line 408 of file GenVertex.h.

References position().

Referenced by HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_production_vertex(), and HepMC::IO_HEPEVT::build_production_vertex().

8.14.4.36 bool HepMC::GenVertex::suggest_barcode (int *the_bar_code*)

In general there is no reason to "suggest_barcode".

allows a barcode to be suggested for this vertex. In general it is better to let the event pick the barcode for you, which is automatic. Returns TRUE if the suggested barcode has been accepted (i.e. the suggested barcode has not already been used in the event, and so it was used). Returns FALSE if the suggested barcode was rejected, or if the vertex is not yet part of an event, such that it is not yet possible to know if the suggested barcode will be accepted).

Definition at line 363 of file GenVertex.cc.

References parent_event(), HepMC::GenEvent::set_barcode(), and set_barcode_().

Referenced by GenVertex().

8.14.4.37 void HepMC::GenVertex::swap (GenVertex & *other*)

swap

Definition at line 71 of file GenVertex.cc.

References m_barcode, m_event, m_id, m_particles_in, m_particles_out, m_position, m_weights, HepMC::WeightContainer::swap(), and HepMC::FourVector::swap().

Referenced by operator=().

8.14.4.38 GenVertex::vertex_iterator HepMC::GenVertex::vertices_begin (IteratorRange *range* = relatives) [inline]

begin vertex range

Definition at line 488 of file GenVertex.h.

References vertex_iterator.

8.14.4.39 GenVertex::vertex_iterator HepMC::GenVertex::vertices_end (IteratorRange) [inline]

end vertex range

Definition at line 494 of file GenVertex.h.

References vertex_iterator.

8.14.4.40 const WeightContainer & HepMC::GenVertex::weights () const [inline]

const direct access to the weights container

Definition at line 405 of file GenVertex.h.

8.14.4.41 WeightContainer & HepMC::GenVertex::weights () [inline]

direct access to the weights container is allowed.

Definition at line 403 of file GenVertex.h.

Referenced by print().

8.14.5 Friends And Related Function Documentation

8.14.5.1 friend class edge_iterator [friend]

Definition at line 217 of file GenVertex.h.

8.14.5.2 friend class GenEvent [friend]

Definition at line 51 of file GenVertex.h.

8.14.5.3 std::ostream& operator<< (std::ostream & *ostr*, const GenVertex & *vtx*) [friend]

print vertex information

Definition at line 440 of file GenVertex.cc.

8.14.5.4 friend class particle_iterator [friend]

Definition at line 350 of file GenVertex.h.

Referenced by particles_begin(), and particles_end().

8.14.5.5 friend class vertex_iterator [friend]

Definition at line 302 of file GenVertex.h.

Referenced by vertices_begin(), and vertices_end().

The documentation for this class was generated from the following files:

- **GenVertex.h**
- **GenVertex.cc**

8.15 HepMC::GenVertex::edge_iterator Class Reference

edge iterator

```
#include <GenVertex.h>
```

Public Member Functions

- **edge_iterator ()**
- **edge_iterator (const GenVertex &vtx, IteratorRange range=family)**
used to set limits on the iteration
- **edge_iterator (const edge_iterator &p)**
copy
- **virtual ~edge_iterator ()**
- **edge_iterator & operator= (const edge_iterator &p)**
make a copy
- **GenParticle * operator * (void) const**
return a pointer to a particle
- **edge_iterator & operator++ (void)**
Pre-fix increment.
- **edge_iterator operator++ (int)**
Post-fix increment.
- **bool operator== (const edge_iterator &a) const**
equality
- **bool operator!= (const edge_iterator &a) const**
inequality
- **bool is_parent () const**
true if parent of root vtx
- **bool is_child () const**
true if child of root vtx
- **const GenVertex * vertex_root () const**
root vertex of this iteration

8.15.1 Detailed Description

edge iterator

iterate over the family of edges connected to m_vertex begins with parents (incoming particles) then children (outgoing) This is not a recursive iterator ... it is a building block for the public iterators and is intended for internal use only. The acceptable Iterator Ranges are: family, parents, children

Definition at line 178 of file GenVertex.h.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 HepMC::GenVertex::edge_iterator::edge_iterator ()

Definition at line 462 of file GenVertex.cc.

8.15.2.2 HepMC::GenVertex::edge_iterator::edge_iterator (const GenVertex & vtx, IteratorRange range = family)

used to set limits on the iteration

Definition at line 466 of file GenVertex.cc.

References HepMC::ancestors, HepMC::children, HepMC::descendants, HepMC::family, HepMC::GenVertex::m_particles_in, HepMC::GenVertex::m_particles_out, and HepMC::parents.

8.15.2.3 HepMC::GenVertex::edge_iterator::edge_iterator (const edge_iterator & p)

copy

Definition at line 517 of file GenVertex.cc.

References p.

8.15.2.4 HepMC::GenVertex::edge_iterator::~~edge_iterator () [virtual]

Definition at line 521 of file GenVertex.cc.

8.15.3 Member Function Documentation

8.15.3.1 bool HepMC::GenVertex::edge_iterator::is_child () const

true if child of root vtx

Definition at line 590 of file GenVertex.cc.

8.15.3.2 bool HepMC::GenVertex::edge_iterator::is_parent () const

true if parent of root vtx

Definition at line 585 of file GenVertex.cc.

Referenced by HepMC::GenVertex::particle_iterator::advance_to_first_(), and HepMC::GenVertex::vertex_iterator::follow_edge_().

8.15.3.3 GenParticle * HepMC::GenVertex::edge_iterator::operator * (void) const

return a pointer to a particle

Definition at line 533 of file GenVertex.cc.

8.15.3.4 `bool HepMC::GenVertex::edge_iterator::operator!=(const edge_iterator & a) const`
[inline]

inequality

Definition at line 451 of file GenVertex.h.

8.15.3.5 `GenVertex::edge_iterator HepMC::GenVertex::edge_iterator::operator++(int)`

Post-fix increment.

Definition at line 578 of file GenVertex.cc.

8.15.3.6 `GenVertex::edge_iterator & HepMC::GenVertex::edge_iterator::operator++(void)`

Pre-fix increment.

Definition at line 538 of file GenVertex.cc.

References HepMC::family, HepMC::GenVertex::m_particles_in, HepMC::GenVertex::m_particles_out, and HepMC::parents.

8.15.3.7 `GenVertex::edge_iterator & HepMC::GenVertex::edge_iterator::operator=(const edge_iterator & p)`

make a copy

Definition at line 523 of file GenVertex.cc.

References p.

8.15.3.8 `bool HepMC::GenVertex::edge_iterator::operator==(const edge_iterator & a) const`
[inline]

equality

Definition at line 446 of file GenVertex.h.

8.15.3.9 `const GenVertex * HepMC::GenVertex::edge_iterator::vertex_root() const` [inline]

root vertex of this iteration

Definition at line 456 of file GenVertex.h.

The documentation for this class was generated from the following files:

- GenVertex.h
- GenVertex.cc

8.16 HepMC::GenVertex::particle_iterator Class Reference

particle iterator

```
#include <GenVertex.h>
```

Public Member Functions

- **particle_iterator ()**
- **particle_iterator (GenVertex &vertex_root, IteratorRange range)**
used to set limits on the iteration
- **particle_iterator (const particle_iterator &)**
copy
- **virtual ~particle_iterator ()**
- **particle_iterator & operator= (const particle_iterator &)**
make a copy
- **GenParticle * operator * (void) const**
return a pointer to a particle
- **particle_iterator & operator++ (void)**
Pre-fix increment.
- **particle_iterator operator++ (int)**
Post-fix increment.
- **bool operator== (const particle_iterator &) const**
equality
- **bool operator!= (const particle_iterator &) const**
inequality

Protected Member Functions

- **GenParticle * advance_to_first_ ()**
"first" particle

8.16.1 Detailed Description

particle iterator

Iterates over all particles connected via a graph. by iterating through all vertices in the m_range. For each vertex it returns orphaned parent particles (i.e. parents without production vertices) then children ... in this way each particle is associated to exactly one vertex and so it is returned exactly once. Is made friend so that it can access protected edge iterator

Examples:

example_UsingIterators.cc, and testHepMCIteration.cc.in.

Definition at line 323 of file GenVertex.h.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 HepMC::GenVertex::particle_iterator::particle_iterator ()

Definition at line 838 of file GenVertex.cc.

8.16.2.2 HepMC::GenVertex::particle_iterator::particle_iterator (GenVertex & *vertex_root*, IteratorRange *range*)

used to set limits on the iteration

Definition at line 840 of file GenVertex.cc.

References `advance_to_first_()`, `HepMC::family`, and `HepMC::GenVertex::vertex_iterator::range()`.

8.16.2.3 HepMC::GenVertex::particle_iterator::particle_iterator (const particle_iterator &)

copy

Definition at line 854 of file GenVertex.cc.

8.16.2.4 HepMC::GenVertex::particle_iterator::~~particle_iterator () [virtual]

Definition at line 859 of file GenVertex.cc.

8.16.3 Member Function Documentation

8.16.3.1 GenParticle * HepMC::GenVertex::particle_iterator::advance_to_first_ () [protected]

"first" particle

if the current edge is not a suitable return value (because it is a parent of the vertex root that itself belongs to a different vertex) it advances to the first suitable return value

Definition at line 900 of file GenVertex.cc.

References `HepMC::GenVertex::edge_iterator::is_parent()`, `HepMC::GenVertex::vertex_iterator::range()`, and `HepMC::relatives`.

Referenced by `operator++()`, and `particle_iterator()`.

8.16.3.2 GenParticle * HepMC::GenVertex::particle_iterator::operator * (void) const

return a pointer to a particle

Definition at line 869 of file GenVertex.cc.

8.16.3.3 `bool HepMC::GenVertex::particle_iterator::operator!=(const particle_iterator &) const` [inline]

inequality

Definition at line 504 of file GenVertex.h.

8.16.3.4 `GenVertex::particle_iterator HepMC::GenVertex::particle_iterator::operator++(int)`

Post-fix increment.

Definition at line 893 of file GenVertex.cc.

8.16.3.5 `GenVertex::particle_iterator & HepMC::GenVertex::particle_iterator::operator++(void)`

Pre-fix increment.

Definition at line 874 of file GenVertex.cc.

References `advance_to_first_()`, and `HepMC::GenVertex::vertex_iterator::range()`.

8.16.3.6 `GenVertex::particle_iterator & HepMC::GenVertex::particle_iterator::operator=(const particle_iterator &)`

make a copy

Definition at line 862 of file GenVertex.cc.

References `m_edge`, and `m_vertex_iterator`.

8.16.3.7 `bool HepMC::GenVertex::particle_iterator::operator==(const particle_iterator &) const` [inline]

equality

Definition at line 499 of file GenVertex.h.

The documentation for this class was generated from the following files:

- `GenVertex.h`
- `GenVertex.cc`

8.17 HepMC::GenVertex::vertex_iterator Class Reference

vertex iterator

```
#include <GenVertex.h>
```

Public Member Functions

- **vertex_iterator ()**
- **vertex_iterator (GenVertex &vtx_root, IteratorRange range)**
used to set limits on the iteration
- **vertex_iterator (GenVertex &vtx_root, IteratorRange range, std::set< const HepMC::GenVertex * > &visited_vertices)**
next constructor is intended for internal use only
- **vertex_iterator (const vertex_iterator &v_iter)**
copy
- **virtual ~vertex_iterator ()**
- **vertex_iterator & operator= (const vertex_iterator &)**
make a copy
- **GenVertex * operator * (void) const**
return a pointer to a vertex
- **vertex_iterator & operator++ (void)**
Pre-fix increment.
- **vertex_iterator operator++ (int)**
Post-fix increment.
- **bool operator== (const vertex_iterator &) const**
equality
- **bool operator!= (const vertex_iterator &) const**
inequality
- **GenVertex * vertex_root () const**
vertex that this iterator begins from
- **IteratorRange range () const**
iterator range
- **void copy_with_own_set (const vertex_iterator &v_iter, std::set< const HepMC::GenVertex * > &visited_vertices)**
intended for internal use only.

Protected Member Functions

- **GenVertex * follow_edge_ ()**
non-null if recursive iter. created
- **void copy_recursive_iterator_ (const vertex_iterator *recursive_v_iter)**
copy recursive iterator

8.17.1 Detailed Description

vertex iterator

Iterates over all vertices connected via a graph to this vertex. this is made friend to that it can access protected edge iterator the range can be IteratorRange= (parents, children, family, ancestors, descendants, relatives) example for range=descendants the iterator will return all vertices which are children (connected by an outgoing particle edge), grandchildren, great-grandchildren, etc. of this vertex In all cases the iterator always returns this vertex (returned last). The algorithm is accomplished by converting the graph to a tree (by "chopping" the edges connecting to an already visited vertex) and returning the vertices in POST ORDER traversal.

Definition at line 247 of file GenVertex.h.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 HepMC::GenVertex::vertex_iterator::vertex_iterator ()

Definition at line 607 of file GenVertex.cc.

Referenced by copy_recursive_iterator_(), and follow_edge_().

8.17.2.2 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & vtx_root, IteratorRange range)

used to set limits on the iteration

Definition at line 612 of file GenVertex.cc.

References HepMC::GenVertex::edges_begin(), HepMC::GenVertex::edges_end(), and follow_edge_().

8.17.2.3 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & vtx_root, IteratorRange range, std::set< const HepMC::GenVertex * > & visited_vertices)

next constructor is intended for internal use only

Definition at line 628 of file GenVertex.cc.

References HepMC::GenVertex::edges_begin(), HepMC::GenVertex::edges_end(), and follow_edge_().

8.17.2.4 HepMC::GenVertex::vertex_iterator::vertex_iterator (const vertex_iterator & v_iter)

copy

Definition at line 645 of file GenVertex.cc.

8.17.2.5 HepMC::GenVertex::vertex_iterator::~~vertex_iterator () [virtual]

Definition at line 652 of file GenVertex.cc.

8.17.3 Member Function Documentation**8.17.3.1 void HepMC::GenVertex::vertex_iterator::copy_recursive_iterator_ (const vertex_iterator * recursive_v_iter)** [protected]

copy recursive iterator

Definition at line 817 of file GenVertex.cc.

References copy_recursive_iterator_(), m_edge, m_it_owns_set, m_range, m_recursive_iterator, m_vertex, m_visited_vertices, and vertex_iterator().

Referenced by copy_recursive_iterator_(), copy_with_own_set(), and operator=().

8.17.3.2 void HepMC::GenVertex::vertex_iterator::copy_with_own_set (const vertex_iterator & v_iter, std::set< const HepMC::GenVertex * > & visited_vertices)

intended for internal use only.

intended for internal use only. (use with care!) this is the same as the operator= method, but it allows the user to specify which set container m_visited_vertices points to. in all cases, this vertex will NOT own its set.

Definition at line 758 of file GenVertex.cc.

References copy_recursive_iterator_(), m_edge, m_range, m_recursive_iterator, and m_vertex.

8.17.3.3 GenVertex * HepMC::GenVertex::vertex_iterator::follow_edge () [protected]

non-null if recursive iter. created

Definition at line 781 of file GenVertex.cc.

References HepMC::family, HepMC::GenVertex::edge_iterator::is_parent(), and vertex_iterator().

Referenced by operator++(), and vertex_iterator().

8.17.3.4 GenVertex * HepMC::GenVertex::vertex_iterator::operator * (void) const

return a pointer to a vertex

Definition at line 694 of file GenVertex.cc.

8.17.3.5 bool HepMC::GenVertex::vertex_iterator::operator!= (const vertex_iterator &) const [inline]

inequality

Definition at line 475 of file GenVertex.h.

8.17.3.6 GenVertex::vertex_iterator HepMC::GenVertex::vertex_iterator::operator++ (int)

Post-fix increment.

Definition at line 751 of file GenVertex.cc.

8.17.3.7 GenVertex::vertex_iterator & HepMC::GenVertex::vertex_iterator::operator++ (void)

Pre-fix increment.

Definition at line 709 of file GenVertex.cc.

References HepMC::GenVertex::edges_end(), and follow_edge_().

8.17.3.8 GenVertex::vertex_iterator & HepMC::GenVertex::vertex_iterator::operator= (const vertex_iterator &)

make a copy

Definition at line 657 of file GenVertex.cc.

References copy_recursive_iterator_(), m_edge, m_it_owns_set, m_range, m_recursive_iterator, m_vertex, and m_visited_vertices.

8.17.3.9 bool HepMC::GenVertex::vertex_iterator::operator== (const vertex_iterator &) const [inline]

equality

Definition at line 470 of file GenVertex.h.

8.17.3.10 IteratorRange HepMC::GenVertex::vertex_iterator::range () const [inline]

iterator range

Definition at line 484 of file GenVertex.h.

Referenced by HepMC::GenVertex::particle_iterator::advance_to_first_(), HepMC::GenVertex::particle_iterator::operator++(), and HepMC::GenVertex::particle_iterator::particle_iterator().

8.17.3.11 GenVertex * HepMC::GenVertex::vertex_iterator::vertex_root () const [inline]

vertex that this iterator begins from

Definition at line 480 of file GenVertex.h.

The documentation for this class was generated from the following files:

- GenVertex.h
- GenVertex.cc

8.18 HepMC::HeavyIon Class Reference

The **HeavyIon** (p. 133) class stores information about heavy ions.

```
#include <HeavyIon.h>
```

Public Member Functions

- **HeavyIon ()**
default constructor
- **HeavyIon (int nh, int np, int nt, int nc, int ns, int nsp, int nnw=0, int nwn=0, int nwnw=0, float im=0., float pl=0., float ec=0., float s=0.)**
The first 6 values must be provided.
- **~HeavyIon ()**
- **HeavyIon (HeavyIon const &orig)**
copy constructor
- **HeavyIon & operator= (HeavyIon const &rhs)**
make a copy
- **void swap (HeavyIon &other)**
swap two HeavyIon (p. 133) objects
- **bool operator== (const HeavyIon &) const**
check for equality
- **bool operator!= (const HeavyIon &) const**
check for inequality
- **int Ncoll_hard () const**
Number of hard scatterings.
- **int Npart_proj () const**
Number of projectile participants.
- **int Npart_targ () const**
Number of target participants.
- **int Ncoll () const**
Number of NN (nucleon-nucleon) collisions.
- **int spectator_neutrons () const**
Number of spectator neutrons.
- **int spectator_protons () const**
Number of spectator protons.
- **int N_Nwounded_collisions () const**

Number of N-Nwounded collisions.

- **int Nwounded_N_collisions () const**
Number of Nwounded-N collisions.
- **int Nwounded_Nwounded_collisions () const**
Number of Nwounded-Nwounded collisions.
- **float impact_parameter () const**
Impact Parameter(in fm) of collision.
- **float event_plane_angle () const**
Azimuthal angle of event plane.
- **float eccentricity () const**
- **float sigma_inel_NN () const**
nucleon-nucleon inelastic (including diffractive) cross-section
- **bool is_valid () const**
verify that the instance contains non-zero information
- **void set_Ncoll_hard (const int &i)**
set number of hard scatterings
- **void set_Npart_proj (const int &i)**
set number of projectile participants
- **void set_Npart_targ (const int &i)**
set number of target participants
- **void set_Ncoll (const int &i)**
set number of NN (nucleon-nucleon) collisions
- **void set_spectator_neutrons (const int &i)**
set number of spectator neutrons
- **void set_spectator_protons (const int &i)**
set number of spectator protons
- **void set_N_Nwounded_collisions (const int &i)**
set number of N-Nwounded collisions
- **void set_Nwounded_N_collisions (const int &i)**
set number of Nwounded-N collisions
- **void set_Nwounded_Nwounded_collisions (const int &i)**
set number of Nwounded-Nwounded collisions
- **void set_impact_parameter (const float &f)**
set Impact Parameter in fm

- **void set_event_plane_angle** (const float &f)
set azimuthal angle of event plane
- **void set_eccentricity** (const float &f)
set eccentricity of participating nucleons in the transverse plane
- **void set_sigma_inel_NN** (const float &f)
set nucleon-nucleon inelastic cross-section

8.18.1 Detailed Description

The **HeavyIon** (p. 133) class stores information about heavy ions.

HepMC::HeavyIon (p. 133) provides additional information storage for Heavy Ion generators in **Gen-Event** (p. 64). Creation and use of this information is optional.

Examples:

`testMass.cc.in.`

Definition at line 45 of file HeavyIon.h.

8.18.2 Constructor & Destructor Documentation

8.18.2.1 HepMC::HeavyIon::HeavyIon () [inline]

default constructor

Definition at line 51 of file HeavyIon.h.

8.18.2.2 HepMC::HeavyIon::HeavyIon (int nh, int np, int nt, int nc, int ns, int nsp, int nnw = 0, int nwn = 0, int nwnw = 0, float im = 0., float pl = 0., float ec = 0., float s = 0.) [inline]

The first 6 values must be provided.

Required members are the number of hard scatterings, the number of projectile participants. the number of target participants. the number of nucleon-nucleon collisions, the number of spectator neutrons, and the number of spectator protons.

Definition at line 178 of file HeavyIon.h.

8.18.2.3 HepMC::HeavyIon::~~HeavyIon () [inline]

Definition at line 72 of file HeavyIon.h.

8.18.2.4 HepMC::HeavyIon::HeavyIon (HeavyIon const & orig) [inline]

copy constructor

Definition at line 196 of file HeavyIon.h.

8.18.3 Member Function Documentation

8.18.3.1 `float HepMC::HeavyIon::eccentricity () const` [inline]

eccentricity of participating nucleons in the transverse plane (as in phobos nucl-ex/0510031)

Definition at line 110 of file HeavyIon.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.18.3.2 `float HepMC::HeavyIon::event_plane_angle () const` [inline]

Azimuthal angle of event plane.

Definition at line 107 of file HeavyIon.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.18.3.3 `float HepMC::HeavyIon::impact_parameter () const` [inline]

Impact Parameter(in fm) of collision.

Definition at line 105 of file HeavyIon.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.18.3.4 `bool HepMC::HeavyIon::is_valid () const` [inline]

verify that the instance contains non-zero information

Definition at line 260 of file HeavyIon.h.

8.18.3.5 `int HepMC::HeavyIon::N_Nwounded_collisions () const` [inline]

Number of N-Nwounded collisions.

Definition at line 99 of file HeavyIon.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.18.3.6 `int HepMC::HeavyIon::Ncoll () const` [inline]

Number of NN (nucleon-nucleon) collisions.

Definition at line 93 of file HeavyIon.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.18.3.7 `int HepMC::HeavyIon::Ncoll_hard () const` [inline]

Number of hard scatterings.

Definition at line 87 of file HeavyIon.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.18.3.8 int HepMC::HeavyIon::Npart_proj () const [inline]

Number of projectile participants.

Definition at line 89 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.9 int HepMC::HeavyIon::Npart_targ () const [inline]

Number of target participants.

Definition at line 91 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.10 int HepMC::HeavyIon::Nwounded_N_collisions () const [inline]

Number of Nwounded-N collisons.

Definition at line 101 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.11 int HepMC::HeavyIon::Nwounded_Nwounded_collisions () const [inline]

Number of Nwounded-Nwounded collisions.

Definition at line 103 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.12 bool HepMC::HeavyIon::operator!= (const HeavyIon &) const [inline]

check for inequality

any nonmatching member generates inequality

Definition at line 254 of file HeavyIon.h.

8.18.3.13 HeavyIon & HepMC::HeavyIon::operator= (HeavyIon const & rhs) [inline]

make a copy

Definition at line 212 of file HeavyIon.h.

References swap().

8.18.3.14 bool HepMC::HeavyIon::operator== (const HeavyIon &) const [inline]

check for equality

equality requires that each member match

Definition at line 236 of file HeavyIon.h.

References `eccentricity()`, `event_plane_angle()`, `impact_parameter()`, `N_Nwounded_collisions()`, `Ncoll()`, `Ncoll_hard()`, `Npart_proj()`, `Npart_targ()`, `Nwounded_N_collisions()`, `Nwounded_Nwounded_collisions()`, `sigma_inel_NN()`, `spectator_neutrons()`, and `spectator_protons()`.

8.18.3.15 `void HepMC::HeavyIon::set_eccentricity (const float &f)` `[inline]`

set eccentricity of participating nucleons in the transverse plane

Definition at line 142 of file `HeavyIon.h`.

Referenced by `HepMC::operator>>()`.

8.18.3.16 `void HepMC::HeavyIon::set_event_plane_angle (const float &f)` `[inline]`

set azimuthal angle of event plane

Definition at line 140 of file `HeavyIon.h`.

Referenced by `HepMC::operator>>()`.

8.18.3.17 `void HepMC::HeavyIon::set_impact_parameter (const float &f)` `[inline]`

set Impact Parameter in fm

Definition at line 138 of file `HeavyIon.h`.

Referenced by `HepMC::operator>>()`.

8.18.3.18 `void HepMC::HeavyIon::set_N_Nwounded_collisions (const int &i)` `[inline]`

set number of N-Nwounded collisions

Definition at line 131 of file `HeavyIon.h`.

Referenced by `HepMC::operator>>()`.

8.18.3.19 `void HepMC::HeavyIon::set_Ncoll (const int &i)` `[inline]`

set number of NN (nucleon-nucleon) collisions

Definition at line 125 of file `HeavyIon.h`.

Referenced by `HepMC::operator>>()`.

8.18.3.20 `void HepMC::HeavyIon::set_Ncoll_hard (const int &i)` `[inline]`

set number of hard scatterings

Definition at line 119 of file `HeavyIon.h`.

Referenced by `HepMC::operator>>()`.

8.18.3.21 `void HepMC::HeavyIon::set_Npart_proj (const int &i)` `[inline]`

set number of projectile participants

Definition at line 121 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.22 void HepMC::HeavyIon::set_Npart_targ (const int & i) [inline]

set number of target participants

Definition at line 123 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.23 void HepMC::HeavyIon::set_Nwounded_N_collisions (const int & i) [inline]

set number of Nwounded-N collisions

Definition at line 133 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.24 void HepMC::HeavyIon::set_Nwounded_Nwounded_collisions (const int & i) [inline]

set number of Nwounded-Nwounded collisions

Definition at line 135 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.25 void HepMC::HeavyIon::set_sigma_inel_NN (const float & f) [inline]

set nucleon-nucleon inelastic cross-section

Definition at line 144 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.26 void HepMC::HeavyIon::set_spectator_neutrons (const int & i) [inline]

set number of spectator neutrons

Definition at line 127 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.27 void HepMC::HeavyIon::set_spectator_protons (const int & i) [inline]

set number of spectator protons

Definition at line 129 of file HeavyIon.h.

Referenced by HepMC::operator>>().

8.18.3.28 float HepMC::HeavyIon::sigma_inel_NN () const [inline]

nucleon-nucleon inelastic (including diffractive) cross-section

Definition at line 112 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.29 `int HepMC::HeavyIon::spectator_neutrons () const` `[inline]`

Number of spectator neutrons.

Definition at line 95 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.30 `int HepMC::HeavyIon::spectator_protons () const` `[inline]`

Number of spectator protons.

Definition at line 97 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

8.18.3.31 `void HepMC::HeavyIon::swap (HeavyIon & other)` `[inline]`

swap two **HeavyIon** (p. 133) objects

Definition at line 219 of file HeavyIon.h.

References `m_eccentricity`, `m_event_plane_angle`, `m_impact_parameter`, `m_N_Nwounded_collisions`, `m_Ncoll`, `m_Ncoll_hard`, `m_Npart_proj`, `m_Npart_targ`, `m_Nwounded_N_collisions`, `m_Nwounded_Nwounded_collisions`, `m_sigma_inel_NN`, `m_spectator_neutrons`, and `m_spectator_protons`.

Referenced by `operator=()`.

The documentation for this class was generated from the following file:

- **HeavyIon.h**

8.19 HepMC::HEPEVT_Wrapper Class Reference

Generic Wrapper for the fortran HEPEVT common block.

```
#include <HEPEVT_Wrapper.h>
```

Static Public Member Functions

- static void **print_hepevt** (std::ostream &ostr=std::cout)
write information from HEPEVT common block
- static void **print_hepevt_particle** (int index, std::ostream &ostr=std::cout)
write particle information to ostr
- static bool **is_double_precision** ()
True if common block uses double.
- static bool **check_hepevt_consistency** (std::ostream &ostr=std::cout)
check for problems with HEPEVT common block
- static void **zero_everything** ()
set all entries in HEPEVT to zero
- static int **event_number** ()
event number
- static int **number_entries** ()
num entries in current evt
- static int **status** (int index)
status code
- static int **id** (int index)
PDG particle id.
- static int **first_parent** (int index)
index of 1st mother
- static int **last_parent** (int index)
index of last mother
- static int **number_parents** (int index)
number of parents
- static int **first_child** (int index)
index of 1st daughter
- static int **last_child** (int index)
index of last daughter

- **static int number_children (int index)**
number of children
- **static double px (int index)**
X momentum.
- **static double py (int index)**
Y momentum.
- **static double pz (int index)**
Z momentum.
- **static double e (int index)**
Energy.
- **static double m (int index)**
generated mass
- **static double x (int index)**
X Production vertex.
- **static double y (int index)**
Y Production vertex.
- **static double z (int index)**
Z Production vertex.
- **static double t (int index)**
production time
- **static void set_event_number (int evtno)**
set event number
- **static void set_number_entries (int noentries)**
set number of entries in HEPEVT
- **static void set_status (int index, int status)**
set particle status
- **static void set_id (int index, int id)**
set particle ID
- **static void set_parents (int index, int firstparent, int lastparent)**
define parents of a particle
- **static void set_children (int index, int firstchild, int lastchild)**
define children of a particle
- **static void set_momentum (int index, double px, double py, double pz, double e)**
set particle momentum

- **static void set_mass (int index, double mass)**
set particle mass
- **static void set_position (int index, double x, double y, double z, double t)**
set particle production vertex
- **static unsigned int sizeof_int ()**
size of integer in bytes
- **static unsigned int sizeof_real ()**
size of real in bytes
- **static int max_number_entries ()**
size of common block
- **static void set_sizeof_int (unsigned int)**
define size of integer
- **static void set_sizeof_real (unsigned int)**
define size of real
- **static void set_max_number_entries (unsigned int)**
define size of common block

Static Protected Member Functions

- **static double byte_num_to_double (unsigned int)**
navigate a byte array
- **static int byte_num_to_int (unsigned int)**
navigate a byte array
- **static void write_byte_num (double, unsigned int)**
pretend common block is an array of bytes
- **static void write_byte_num (int, unsigned int)**
pretend common block is an array of bytes
- **static void print_legend (std::ostream &ostr=std::cout)**
print output legend

8.19.1 Detailed Description

Generic Wrapper for the fortran HEPEVT common block.

This class is intended for static use only - it makes no sense to instantiate it.

Definition at line 130 of file HEPEVT_Wrapper.h.

8.19.2 Member Function Documentation

8.19.2.1 `double HepMC::HEPEVT_Wrapper::byte_num_to_double (unsigned int)` `[inline, static, protected]`

navigate a byte array

Definition at line 255 of file HEPEVT_Wrapper.h.

References `hepevt`, and `hepevt_bytes_allocation`.

Referenced by `e()`, `m()`, `px()`, `py()`, `pz()`, `t()`, `x()`, `y()`, and `z()`.

8.19.2.2 `int HepMC::HEPEVT_Wrapper::byte_num_to_int (unsigned int)` `[inline, static, protected]`

navigate a byte array

Definition at line 273 of file HEPEVT_Wrapper.h.

References `hepevt`, and `hepevt_bytes_allocation`.

Referenced by `event_number()`, `first_child()`, `first_parent()`, `id()`, `last_child()`, `last_parent()`, `number_entries()`, and `status()`.

8.19.2.3 `bool HepMC::HEPEVT_Wrapper::check_hepevt_consistency (std::ostream & ostr = std::cout)` `[static]`

check for problems with HEPEVT common block

This method inspects the HEPEVT common block and looks for inconsistencies in the mother/daughter pointers

Definition at line 88 of file HEPEVT_Wrapper.cc.

References `event_number()`, `first_child()`, `first_parent()`, `last_child()`, `last_parent()`, `m()`, `number_entries()`, `print_hepevt_particle()`, and `print_legend()`.

8.19.2.4 `double HepMC::HEPEVT_Wrapper::e (int index)` `[inline, static]`

Energy.

Definition at line 446 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_particle()`, `HepMC::IO_HEPEVT::build_particle()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.5 `int HepMC::HEPEVT_Wrapper::event_number ()` `[inline, static]`

event number

Definition at line 343 of file HEPEVT_Wrapper.h.

References `byte_num_to_int()`.

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `HepMC::IO_HERWIG::build_production_vertex()`, `HepMC::IO_HEPEVT::build_production_vertex()`,

check_hepevt_consistency(), HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), and print_hepevt().

8.19.2.6 int HepMC::HEPEVT_Wrapper::first_child (int *index*) [inline, static]

index of 1st daughter

Definition at line 394 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_end_vertex(), check_hepevt_consistency(), last_child(), number_children(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.19.2.7 int HepMC::HEPEVT_Wrapper::first_parent (int *index*) [inline, static]

index of 1st mother

Definition at line 362 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), check_hepevt_consistency(), HepMC::IO_HERWIG::fill_next_event(), last_parent(), number_parents(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.19.2.8 int HepMC::HEPEVT_Wrapper::id (int *index*) [inline, static]

PDG particle id.

Definition at line 356 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), and sizeof_int().

Referenced by HepMC::IO_HERWIG::build_particle(), HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.19.2.9 bool HepMC::HEPEVT_Wrapper::is_double_precision () [inline, static]

True if common block uses double.

Definition at line 337 of file HEPEVT_Wrapper.h.

References sizeof_real().

Referenced by print_hepevt().

8.19.2.10 int HepMC::HEPEVT_Wrapper::last_child (int *index*) [inline, static]

index of last daughter

Definition at line 402 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), first_child(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `check_hepevt_consistency()`, `number_children()`, `print_hepevt_particle()`, `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, and `HepMC::IO_HERWIG::repair_hepevt()`.

8.19.2.11 `int HepMC::HEPEVT_Wrapper::last_parent (int index)` `[inline, static]`

index of last mother

Definition at line 370 of file `HEPEVT_Wrapper.h`.

References `byte_num_to_int()`, `first_parent()`, `max_number_entries()`, `number_entries()`, and `sizeof_int()`.

Referenced by `HepMC::IO_HERWIG::build_production_vertex()`, `HepMC::IO_HEPEVT::build_production_vertex()`, `check_hepevt_consistency()`, `number_parents()`, `print_hepevt_particle()`, `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, and `HepMC::IO_HERWIG::repair_hepevt()`.

8.19.2.12 `double HepMC::HEPEVT_Wrapper::m (int index)` `[inline, static]`

generated mass

Definition at line 452 of file `HEPEVT_Wrapper.h`.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_particle()`, `HepMC::IO_HEPEVT::build_particle()`, `check_hepevt_consistency()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.13 `int HepMC::HEPEVT_Wrapper::max_number_entries ()` `[inline, static]`

size of common block

Definition at line 229 of file `HEPEVT_Wrapper.h`.

Referenced by `e()`, `first_child()`, `first_parent()`, `id()`, `last_child()`, `last_parent()`, `m()`, `number_entries()`, `print_hepevt()`, `px()`, `py()`, `pz()`, `set_children()`, `set_id()`, `set_mass()`, `set_momentum()`, `set_parents()`, `set_position()`, `set_status()`, `t()`, `HepMC::IO_HEPEVT::write_event()`, `x()`, `y()`, `z()`, `zero_everything()`, and `HepMC::IO_HERWIG::zero_hepevt_entry()`.

8.19.2.14 `int HepMC::HEPEVT_Wrapper::number_children (int index)` `[inline, static]`

number of children

Definition at line 420 of file `HEPEVT_Wrapper.h`.

References `first_child()`, and `last_child()`.

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, and `HepMC::IO_HEPEVT::build_end_vertex()`.

8.19.2.15 `int HepMC::HEPEVT_Wrapper::number_entries ()` `[inline, static]`

num entries in current evt

Definition at line 346 of file `HEPEVT_Wrapper.h`.

References `byte_num_to_int()`, `max_number_entries()`, and `sizeof_int()`.

Referenced by `check_hepevt_consistency()`, `HepMC::IO_HERWIG::fill_next_event()`, `HepMC::IO_HEPEVT::fill_next_event()`, `first_child()`, `first_parent()`, `last_child()`, `last_parent()`, `print_hepevt()`, `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, and `HepMC::IO_HERWIG::repair_hepevt()`.

8.19.2.16 `int HepMC::HEPEVT_Wrapper::number_parents (int index)` `[inline, static]`

number of parents

Definition at line 388 of file `HEPEVT_Wrapper.h`.

References `first_parent()`, and `last_parent()`.

Referenced by `HepMC::IO_HERWIG::build_production_vertex()`, and `HepMC::IO_HEPEVT::build_production_vertex()`.

8.19.2.17 `void HepMC::HEPEVT_Wrapper::print_hepevt (std::ostream & ostr = std::cout)` `[static]`

write information from HEPEVT common block

dumps the content of this HEPEVT event to ostr (Width is 80)

Examples:

`example_MyHerwig.cc`.

Definition at line 27 of file `HEPEVT_Wrapper.cc`.

References `event_number()`, `is_double_precision()`, `max_number_entries()`, `number_entries()`, `print_hepevt_particle()`, `print_legend()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `main()`.

8.19.2.18 `void HepMC::HEPEVT_Wrapper::print_hepevt_particle (int index, std::ostream & ostr = std::cout)` `[static]`

write particle information to ostr

dumps the content HEPEVT particle entry i (Width is 120) here i is the C array index (i.e. it starts at 0 ... whereas the fortran array index starts at 1) So if there's 100 particles, the last valid index is 100-1=99

Definition at line 68 of file `HEPEVT_Wrapper.cc`.

References `e()`, `first_child()`, `first_parent()`, `last_child()`, `last_parent()`, `m()`, `px()`, `py()`, `pz()`, `status()`, `t()`, `x()`, `y()`, and `z()`.

Referenced by `check_hepevt_consistency()`, and `print_hepevt()`.

8.19.2.19 `void HepMC::HEPEVT_Wrapper::print_legend (std::ostream & ostr = std::cout)` `[static, protected]`

print output legend

Definition at line 55 of file `HEPEVT_Wrapper.cc`.

Referenced by `check_hepevt_consistency()`, and `print_hepevt()`.

8.19.2.20 double HepMC::HEPEVT_Wrapper::px (int *index*) [inline, static]

X momentum.

Definition at line 427 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_particle()`, `HepMC::IO_HEPEVT::build_particle()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.21 double HepMC::HEPEVT_Wrapper::py (int *index*) [inline, static]

Y momentum.

Definition at line 433 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_particle()`, `HepMC::IO_HEPEVT::build_particle()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.22 double HepMC::HEPEVT_Wrapper::pz (int *index*) [inline, static]

Z momentum.

Definition at line 440 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_particle()`, `HepMC::IO_HEPEVT::build_particle()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.23 void HepMC::HEPEVT_Wrapper::set_children (int *index*, int *firstchild*, int *lastchild*)
[inline, static]

define children of a particle

Definition at line 514 of file HEPEVT_Wrapper.h.

References `max_number_entries()`, `sizeof_int()`, and `write_byte_num()`.

Referenced by `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, `HepMC::IO_HERWIG::repair_hepevt()`, `HepMC::IO_HEPEVT::write_event()`, `zero_everything()`, and `HepMC::IO_HERWIG::zero_hepevt_entry()`.

8.19.2.24 void HepMC::HEPEVT_Wrapper::set_event_number (int *evtno*) [inline, static]

set event number

Definition at line 486 of file HEPEVT_Wrapper.h.

References `write_byte_num()`.

Referenced by `HepMC::IO_HEPEVT::write_event()`, and `zero_everything()`.

8.19.2.25 void HepMC::HEPEVT_Wrapper::set_id (int *index*, int *id*) [inline, static]

set particle ID

Definition at line 498 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HERWIG::repair_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

8.19.2.26 void HepMC::HEPEVT_Wrapper::set_mass (int *index*, double *mass*) [inline, static]

set particle mass

Definition at line 538 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), sizeof_real(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

8.19.2.27 void HepMC::HEPEVT_Wrapper::set_max_number_entries (unsigned *int*) [inline, static]

define size of common block

Examples:

example_MyHerwig.cc, example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc, testHerwigCopies.cc, and testPythiaCopies.cc.

Definition at line 251 of file HEPEVT_Wrapper.h.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and write_PythiaStreamIO().

8.19.2.28 void HepMC::HEPEVT_Wrapper::set_momentum (int *index*, double *px*, double *py*, double *pz*, double *e*) [inline, static]

set particle momentum

Definition at line 524 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), sizeof_real(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

8.19.2.29 void HepMC::HEPEVT_Wrapper::set_number_entries (int *noentries*) [inline, static]

set number of entries in HEPEVT

Definition at line 489 of file HEPEVT_Wrapper.h.

References `sizeof_int()`, and `write_byte_num()`.

Referenced by `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, `HepMC::IO_HEPEVT::write_event()`, and `zero_everything()`.

8.19.2.30 `void HepMC::HEPEVT_Wrapper::set_parents (int index, int firstparent, int lastparent)` `[inline, static]`

define parents of a particle

Definition at line 504 of file `HEPEVT_Wrapper.h`.

References `max_number_entries()`, `sizeof_int()`, and `write_byte_num()`.

Referenced by `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, `HepMC::IO_HERWIG::repair_hepevt()`, `HepMC::IO_HEPEVT::write_event()`, `zero_everything()`, and `HepMC::IO_HERWIG::zero_hepevt_entry()`.

8.19.2.31 `void HepMC::HEPEVT_Wrapper::set_position (int index, double x, double y, double z, double t)` `[inline, static]`

set particle production vertex

Definition at line 545 of file `HEPEVT_Wrapper.h`.

References `max_number_entries()`, `sizeof_int()`, `sizeof_real()`, and `write_byte_num()`.

Referenced by `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, `HepMC::IO_HEPEVT::write_event()`, `zero_everything()`, and `HepMC::IO_HERWIG::zero_hepevt_entry()`.

8.19.2.32 `void HepMC::HEPEVT_Wrapper::set_sizeof_int (unsigned int)` `[inline, static]`

define size of integer

Definition at line 232 of file `HEPEVT_Wrapper.h`.

8.19.2.33 `void HepMC::HEPEVT_Wrapper::set_sizeof_real (unsigned int)` `[inline, static]`

define size of real

Examples:

`example_MyHerwig.cc`, `example_MyPythia.cc`, `example_MyPythiaOnlyToHepMC.cc`, `example_PythiaStreamIO.cc`, `testHerwigCopies.cc`, and `testPythiaCopies.cc`.

Definition at line 242 of file `HEPEVT_Wrapper.h`.

Referenced by `event_selection()`, `main()`, `pythia_in_out()`, `pythia_out()`, `pythia_particle_out()`, and `write_PythiaStreamIO()`.

8.19.2.34 `void HepMC::HEPEVT_Wrapper::set_status (int index, int status)` `[inline, static]`

set particle status

Definition at line 492 of file HEPEVT_Wrapper.h.

References `max_number_entries()`, `sizeof_int()`, and `write_byte_num()`.

Referenced by `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, `HepMC::IO_HEPEVT::write_event()`, `zero_everything()`, and `HepMC::IO_HERWIG::zero_hepevt_entry()`.

8.19.2.35 unsigned int HepMC::HEPEVT_Wrapper::sizeof_int () [inline, static]

size of integer in bytes

Definition at line 225 of file HEPEVT_Wrapper.h.

Referenced by `e()`, `first_child()`, `first_parent()`, `id()`, `last_child()`, `last_parent()`, `m()`, `number_entries()`, `print_hepevt()`, `px()`, `py()`, `pz()`, `set_children()`, `set_id()`, `set_mass()`, `set_momentum()`, `set_number_entries()`, `set_parents()`, `set_position()`, `set_status()`, `status()`, `t()`, `x()`, `y()`, and `z()`.

8.19.2.36 unsigned int HepMC::HEPEVT_Wrapper::sizeof_real () [inline, static]

size of real in bytes

Definition at line 227 of file HEPEVT_Wrapper.h.

Referenced by `e()`, `is_double_precision()`, `m()`, `print_hepevt()`, `px()`, `py()`, `pz()`, `set_mass()`, `set_momentum()`, `set_position()`, `t()`, `x()`, `y()`, and `z()`.

8.19.2.37 int HepMC::HEPEVT_Wrapper::status (int index) [inline, static]

status code

Definition at line 353 of file HEPEVT_Wrapper.h.

References `byte_num_to_int()`, and `sizeof_int()`.

Referenced by `HepMC::IO_HERWIG::build_particle()`, `HepMC::IO_HEPEVT::build_particle()`, `HepMC::IO_HERWIG::fill_next_event()`, `print_hepevt_particle()`, `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`, and `HepMC::IO_HERWIG::repair_hepevt()`.

8.19.2.38 double HepMC::HEPEVT_Wrapper::t (int index) [inline, static]

production time

Definition at line 479 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `HepMC::IO_HERWIG::build_production_vertex()`, `HepMC::IO_HEPEVT::build_production_vertex()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.39 void HepMC::HEPEVT_Wrapper::write_byte_num (int, unsigned int) [inline, static, protected]

pretend common block is an array of bytes

Definition at line 312 of file HEPEVT_Wrapper.h.

References `hepevt`, and `hepevt_bytes_allocation`.

8.19.2.40 `void HepMC::HEPEVT_Wrapper::write_byte_num (double, unsigned int)` [inline, static, protected]

pretend common block is an array of bytes

Definition at line 295 of file HEPEVT_Wrapper.h.

References `hepevt`, and `hepevt_bytes_allocation`.

Referenced by `set_children()`, `set_event_number()`, `set_id()`, `set_mass()`, `set_momentum()`, `set_number_entries()`, `set_parents()`, `set_position()`, and `set_status()`.

8.19.2.41 `double HepMC::HEPEVT_Wrapper::x (int index)` [inline, static]

X Production vertex.

Definition at line 458 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `HepMC::IO_HERWIG::build_production_vertex()`, `HepMC::IO_HEPEVT::build_production_vertex()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.42 `double HepMC::HEPEVT_Wrapper::y (int index)` [inline, static]

Y Production vertex.

Definition at line 465 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `HepMC::IO_HERWIG::build_production_vertex()`, `HepMC::IO_HEPEVT::build_production_vertex()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.43 `double HepMC::HEPEVT_Wrapper::z (int index)` [inline, static]

Z Production vertex.

Definition at line 472 of file HEPEVT_Wrapper.h.

References `byte_num_to_double()`, `max_number_entries()`, `sizeof_int()`, and `sizeof_real()`.

Referenced by `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `HepMC::IO_HERWIG::build_production_vertex()`, `HepMC::IO_HEPEVT::build_production_vertex()`, `print_hepevt_particle()`, and `HepMC::IO_HERWIG::remove_gaps_in_hepevt()`.

8.19.2.44 `void HepMC::HEPEVT_Wrapper::zero_everything ()` [static]

set all entries in HEPEVT to zero

Definition at line 212 of file HEPEVT_Wrapper.cc.

References `max_number_entries()`, `set_children()`, `set_event_number()`, `set_id()`, `set_mass()`, `set_momentum()`, `set_number_entries()`, `set_parents()`, `set_position()`, and `set_status()`.

The documentation for this class was generated from the following files:

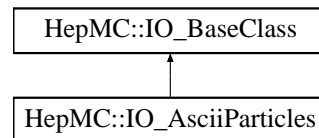
- **HEPEVT_Wrapper.h**
- **HEPEVT_Wrapper.cc**

8.20 HepMC::IO_AsciiParticles Class Reference

event input/output in ascii format for eye and machine reading

```
#include <IO_AsciiParticles.h>
```

Inheritance diagram for HepMC::IO_AsciiParticles::



Public Member Functions

- **IO_AsciiParticles** (const char *filename="IO_AsciiParticles.dat", std::ios::openmode mode=std::ios::out)
constructor requiring a file name and std::ios mode
- **virtual ~IO_AsciiParticles** ()
- **void write_event** (const GenEvent *evt)
write this event
- **bool fill_next_event** (GenEvent *evt)
get the next event
- **void write_particle_data_table** (const ParticleDataTable *)
write this ParticleDataTable (p. 211)
- **bool fill_particle_data_table** (ParticleDataTable *)
fill this ParticleDataTable (p. 211)
- **void write_comment** (const std::string comment)
- **void setPrecision** (int iprec)
set output precision
- **int rdstate** () const
check the state of the IO stream
- **void clear** ()
clear the IO stream
- **void print** (std::ostream &ostr=std::cout) const
write to ostr

Protected Member Functions

- **bool write_end_listing** ()
write end tag

8.20.1 Detailed Description

event input/output in ascii format for eye and machine reading

Strategy for reading or writing events/particleData as machine readable ascii to a file. When instantiating, the mode of file to be created must be specified.

Examples:

example_MyPythia.cc, testHepMC.cc.in, and testStreamIO.cc.in.

Definition at line 54 of file IO_AsciiParticles.h.

8.20.2 Constructor & Destructor Documentation

8.20.2.1 HepMC::IO_AsciiParticles::IO_AsciiParticles (const char * *filename* = "IO_AsciiParticles.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

Definition at line 18 of file IO_AsciiParticles.cc.

8.20.2.2 HepMC::IO_AsciiParticles::~~IO_AsciiParticles () [virtual]

Definition at line 47 of file IO_AsciiParticles.cc.

8.20.3 Member Function Documentation

8.20.3.1 void HepMC::IO_AsciiParticles::clear () [inline]

clear the IO stream

Definition at line 99 of file IO_AsciiParticles.h.

8.20.3.2 bool HepMC::IO_AsciiParticles::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** **p.** (classHepMC₁₁*IO_BaseClass_1dfb95a44d521af510f6431a30f942* ??)

Definition at line 180 of file IO_AsciiParticles.cc.

8.20.3.3 bool HepMC::IO_AsciiParticles::fill_particle_data_table (ParticleDataTable *) [inline, virtual]

fill this **ParticleDataTable** (p.211)

Implements **HepMC::IO_BaseClass** **p.** (classHepMC₁₁*IO_BaseClass_94fc72dd621a2283989525497715feb* ??)

Definition at line 107 of file IO_AsciiParticles.h.

8.20.3.4 `void HepMC::IO_AsciiParticles::print (std::ostream & ostr = std::cout) const` `[virtual]`

write to ostr

Reimplemented from **HepMC::IO_BaseClass** `p.` (classHepMC₁₁*IO_BaseClass*_{8a23f5de9c6bb10931dcacdeb7677413}??)

Definition at line 54 of file IO_AsciiParticles.cc.

8.20.3.5 `int HepMC::IO_AsciiParticles::rdstate () const` `[inline]`

check the state of the IO stream

Definition at line 98 of file IO_AsciiParticles.h.

8.20.3.6 `void HepMC::IO_AsciiParticles::setPrecision (int iprec)` `[inline]`

set output precision

Definition at line 100 of file IO_AsciiParticles.h.

8.20.3.7 `void HepMC::IO_AsciiParticles::write_comment (const std::string comment)`

insert a comment directly into the output file — normally you only want to do this at the beginning or end of the file. All comments are preceded with "HepMC::IO_AsciiParticles-COMMENT\n"

Definition at line 203 of file IO_AsciiParticles.cc.

References `write_end_listing()`.

8.20.3.8 `bool HepMC::IO_AsciiParticles::write_end_listing ()` `[protected]`

write end tag

Definition at line 218 of file IO_AsciiParticles.cc.

Referenced by `write_comment()`.

8.20.3.9 `void HepMC::IO_AsciiParticles::write_event (const GenEvent * evt)` `[virtual]`

write this event

Implements **HepMC::IO_BaseClass** `p.` (classHepMC₁₁*IO_BaseClass*_{7929dfd8412207c7904f29652810c1f4}??)

Definition at line 64 of file IO_AsciiParticles.cc.

References `HepMC::GenEvent::alphaQCD()`, `HepMC::GenEvent::alphaQED()`, `HepMC::GenVertex::barcode()`, `HepMC::WeightContainer::begin()`, `HepMC::WeightContainer::end()`, `HepMC::GenEvent::event_number()`, `HepMC::GenEvent::event_scale()`, `HepMC::detail::output()`, `HepMC::GenEvent::particles_begin()`, `HepMC::GenEvent::particles_end()`, `HepMC::GenEvent::particles_size()`, `HepMC::GenEvent::random_states()`, `HepMC::GenEvent::signal_process_id()`, `HepMC::GenEvent::signal_process_vertex()`, `HepMC::WeightContainer::size()`, `HepMC::versionName()`, `HepMC::GenEvent::vertices_size()`, and `HepMC::GenEvent::weights()`.

8.20.3.10 void HepMC::IO_AsciiParticles::write_particle_data_table (const ParticleDataTable *)
[inline, virtual]

write this **ParticleDataTable** (p. 211)

Implements **HepMC::IO_BaseClass** **p.** (classHepMC11IO_BaseClass1c2fec4084d50bae9ce6b4cbdf133df4 ??)

Definition at line 106 of file IO_AsciiParticles.h.

The documentation for this class was generated from the following files:

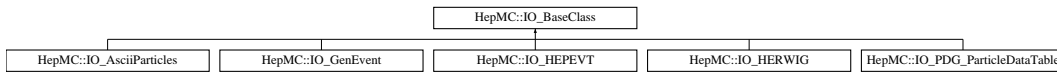
- **IO_AsciiParticles.h**
- **IO_AsciiParticles.cc**

8.21 HepMC::IO_BaseClass Class Reference

all input/output classes inherit from **IO_BaseClass** (p. 158)

```
#include <IO_BaseClass.h>
```

Inheritance diagram for HepMC::IO_BaseClass::



Public Member Functions

- virtual `~IO_BaseClass()`
- virtual void `write_event (const GenEvent *)=0`
write this GenEvent (p. 64)
- virtual bool `fill_next_event (GenEvent *)=0`
fill this GenEvent (p. 64)
- virtual void `write_particle_data_table (const ParticleDataTable *)=0`
write this ParticleDataTable (p. 211)
- virtual bool `fill_particle_data_table (ParticleDataTable *)=0`
fill this ParticleDataTable (p. 211)
- virtual void `print (std::ostream &ostr=std::cout) const`
write output to ostr
- `GenEvent * read_next_event ()`
do not over-ride
- `ParticleDataTable * read_particle_data_table ()`
do not over-ride
- virtual `GenEvent *& operator>> (GenEvent *&)`
the same as read_next_event
- virtual `const GenEvent *& operator<< (const GenEvent *&)`
the same as write_event
- virtual `GenEvent *& operator<< (GenEvent *&)`
the same as write_event
- virtual `ParticleDataTable *& operator>> (ParticleDataTable *&)`
the same as read_particle_data_table
- virtual `const ParticleDataTable *& operator<< (const ParticleDataTable *&)`
the same as write_particle_data_table

- **virtual ParticleDataTable *& operator<< (ParticleDataTable *&)**
the same as write_particle_data_table

8.21.1 Detailed Description

all input/output classes inherit from **IO_BaseClass** (p. 158)

If you want to write a new IO class, then inherit from this class and re-define read_event() and **write_event()** (p. 161)

Definition at line 35 of file IO_BaseClass.h.

8.21.2 Constructor & Destructor Documentation

8.21.2.1 virtual HepMC::IO_BaseClass::~IO_BaseClass () [inline, virtual]

Definition at line 37 of file IO_BaseClass.h.

8.21.3 Member Function Documentation

8.21.3.1 virtual bool HepMC::IO_BaseClass::fill_next_event (GenEvent *) [pure virtual]

fill this **GenEvent** (p. 64)

Implemented in **HepMC::IO_AsciiParticles** p. (classHepMC₁₁*IO_AsciiParticles*_{fd859891c2ac09f8758d081357c17ce}??)HepMC

fill this **ParticleDataTable** (p. 211)

Implemented in **HepMC::IO_AsciiParticles** p. (classHepMC₁₁*IO_AsciiParticles*_{e03b03bffb39bcd419dc2c1a4de0b1d}??)HepMC

the same as write_particle_data_table

Definition at line 149 of file IO_BaseClass.h.

References write_particle_data_table().

8.21.3.4 const ParticleDataTable *& HepMC::IO_BaseClass::operator<< (const ParticleDataTable *&) [inline, virtual]

the same as write_particle_data_table

Definition at line 143 of file IO_BaseClass.h.

References write_particle_data_table().

8.21.3.5 GenEvent *& HepMC::IO_BaseClass::operator<< (GenEvent *&) [inline, virtual]

the same as write_event

Definition at line 132 of file IO_BaseClass.h.

References write_event().

8.21.3.6 `const GenEvent *& HepMC::IO_BaseClass::operator<< (const GenEvent *&)` [inline, virtual]

the same as `write_event`

Definition at line 126 of file `IO_BaseClass.h`.

References `write_event()`.

8.21.3.7 `ParticleDataTable *& HepMC::IO_BaseClass::operator>> (ParticleDataTable *&)` [inline, virtual]

the same as `read_particle_data_table`

Definition at line 137 of file `IO_BaseClass.h`.

References `read_particle_data_table()`.

8.21.3.8 `GenEvent *& HepMC::IO_BaseClass::operator>> (GenEvent *&)` [inline, virtual]

the same as `read_next_event`

Definition at line 121 of file `IO_BaseClass.h`.

References `read_next_event()`.

8.21.3.9 `void HepMC::IO_BaseClass::print (std::ostream & ostr = std::cout) const` [inline, virtual]

write output to `ostr`

Reimplemented in `HepMC::IO_AsciiParticles p.` (classHepMC₁₁*IO_AsciiParticles*_{2c9bec0be07d8b946ff8c27bf2d0636} ??)HepM

do not over-ride

creates a new event and fills it by calling the sister method `read_next_event(GenEvent*)`

Examples:

`example_MyHerwig.cc`, `example_MyPythia.cc`, `example_MyPythiaOnlyToHepMC.cc`,
`example_PythiaStreamIO.cc`, `testHerwigCopies.cc`, `testMultipleCopies.cc.in`, and `testPythia-Copies.cc`.

Definition at line 87 of file `IO_BaseClass.h`.

References `fill_next_event()`.

Referenced by `event_selection()`, `main()`, `operator>>()`, `pythia_in()`, `pythia_in_out()`, `pythia_out()`, `pythia_particle_out()`, and `writePythiaStreamIO()`.

8.21.3.11 `ParticleDataTable * HepMC::IO_BaseClass::read_particle_data_table ()` [inline]

do not over-ride

creates a new particle data table and fills it by calling the sister method `read_particle_data_table(ParticleDataTable*)`

Definition at line 103 of file IO_BaseClass.h.

References `fill_particle_data_table()`.

Referenced by `operator>>()`.

8.21.3.12 `virtual void HepMC::IO_BaseClass::write_event (const GenEvent *)` [pure virtual]

write this **GenEvent** (p. 64)

Implemented in **HepMC::IO_AsciiParticles p.** (classHepMC₁₁*IO_AsciiParticles_{2d45b9474967ec0cdeccece86d5d8c5e}*??)HepMC :

write this **ParticleDataTable** (p. 211)

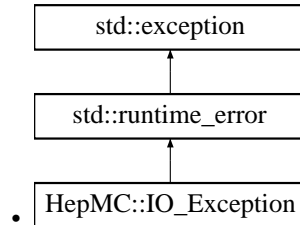
Implemented in **HepMC::IO_AsciiParticles p.** (classHepMC₁₁*IO_AsciiParticles_{94c2db38f83b52f5724777d17399457c}*??)andHepM

8.22 HepMC::IO_Exception Class Reference

IO exception handling.

```
#include <IO_Exception.h>
```

Inheritance diagram for HepMC::IO_Exception::



Public Types

- **OK**
- **NullEvent**
- **WrongFileType**
- **MissingStartKey**
- **EndOfStream**
- **EndKeyMismatch**
- **MissingEndKey**
- **InvalidData**
- **InputAndOutput**
- **BadOutputStream**
- **BadInputStream**
- **enum ErrorType {**
 OK, NullEvent, WrongFileType, MissingStartKey,
 EndOfStream, EndKeyMismatch, MissingEndKey, InvalidData,
 InputAndOutput, BadOutputStream, BadInputStream }
IO error types.

Public Member Functions

- **IO_Exception (const std::string &msg)**

8.22.1 Detailed Description

IO exception handling.

IO_GenEvent (p. 164), etc. catch the throw and set data members with the error type and message Some of the messages are constructed with transient information (e.g., contents of a bad **GenParticle** (p. 99))

Examples:

testStreamIO.cc.in.

Definition at line 28 of file `IO_Exception.h`.

8.22.2 Member Enumeration Documentation

8.22.2.1 enum HepMC::IO_Exception::ErrorType

IO error types.

Enumerator:

OK
NullEvent
WrongFileType
MissingStartKey
EndOfStream
EndKeyMismatch
MissingEndKey
InvalidData
InputAndOutput
BadOutputStream
BadInputStream

Definition at line 34 of file IO_Exception.h.

8.22.3 Constructor & Destructor Documentation

8.22.3.1 HepMC::IO_Exception::IO_Exception (const std::string & msg) [inline]

Definition at line 30 of file IO_Exception.h.

The documentation for this class was generated from the following file:

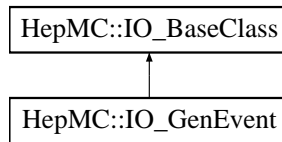
- **IO_Exception.h**

8.23 HepMC::IO_GenEvent Class Reference

IO_GenEvent (p. 164) also deals with **HeavyIon** (p. 133) and **PdfInfo** (p. 218).

```
#include <IO_GenEvent.h>
```

Inheritance diagram for HepMC::IO_GenEvent::



Public Member Functions

- **IO_GenEvent** (const std::string &filename="IO_GenEvent.dat", std::ios::openmode mode=std::ios::out)
constructor requiring a file name and std::ios mode
- **IO_GenEvent** (std::istream &)
constructor requiring an input stream
- **IO_GenEvent** (std::ostream &)
constructor requiring an output stream
- **virtual ~IO_GenEvent** ()
- **void write_event** (const GenEvent *evt)
write this event
- **bool fill_next_event** (GenEvent *evt)
get the next event
- **void write_particle_data_table** (const ParticleDataTable *)
write_particle_data_table is required by IO_BaseClass (p. 158), but not used here
- **bool fill_particle_data_table** (ParticleDataTable *)
fill_particle_data_table is required by IO_BaseClass (p. 158), but not used here
- **void write_comment** (const std::string comment)
- **int rdstate** () const
check the state of the IO stream
- **void clear** ()
clear the IO stream
- **void print** (std::ostream &ostr=std::cout) const
write to ostr
- **void use_input_units** (Units::MomentumUnit, Units::LengthUnit)

- `void precision (int)`
- `const int error_type () const`
integer (enum) associated with read error
- `const std::string & error_message () const`
the read error message string

Protected Member Functions

- `ParticleData * read_particle_data (std::istream *, ParticleDataTable *)`
- `bool read_io_particle_data_table (std::istream *, ParticleDataTable *)`

8.23.1 Detailed Description

IO_GenEvent (p. 164) also deals with **HeavyIon** (p. 133) and **PdfInfo** (p. 218).

event input/output in ascii format for machine reading extended format contains **HeavyIon** (p. 133) and **PdfInfo** (p. 218) classes

Strategy for reading or writing events using iostreams When instantiating with a file name, the mode of file to be created must be specified. Options are: `std::ios::in` open file for input `std::ios::out` open file for output `std::ios::trunc` erase old file when opening (i.e. `ios::out|iostrunc` removes oldfile, and creates a new one for output) `std::ios::app` append output to end of file for the purposes of this class, simultaneous input and output mode (`std::ios::in | std::ios::out`) is not allowed.

Event listings are preceded by the key: "HepMC::IO_GenEvent-START_EVENT_LISTING\n" and terminated by the key: "HepMC::IO_GenEvent-END_EVENT_LISTING\n" **GenParticle** (p. 99) Data tables are preceded by the key: "HepMC::IO_GenEvent-START_PARTICLE_DATA\n" and terminated by the key: "HepMC::IO_GenEvent-END_PARTICLE_DATA\n" Comments are allowed. They need not be preceded by anything, though if a comment is written using `write_comment(const string)` then it will be preceded by "HepMC::IO_GenEvent-COMMENT\n" Each event, vertex, particle, particle data, heavy ion, or pdf info line is preceded by "E ", "V ", "P ", "D ", "H ", "F " respectively. Comments may appear anywhere in the file – so long as they do not contain any of the start/stop keys.

Examples:

`example_EventSelection.cc`, `example_MyHerwig.cc`, `example_MyPythia.cc`, `example_Using-Iterators.cc`, `testFlow.cc`, `testHepMC.cc.in`, `testHepMCIteration.cc.in`, `testMass.cc.in`, `test-MultipleCopies.cc.in`, and `testStreamIO.cc.in`.

Definition at line 64 of file `IO_GenEvent.h`.

8.23.2 Constructor & Destructor Documentation

8.23.2.1 HepMC::IO_GenEvent::IO_GenEvent (const std::string & filename = "IO_GenEvent.dat", std::ios::openmode mode = std::ios::out)

constructor requiring a file name and `std::ios` mode

Definition at line 17 of file `IO_GenEvent.cc`.

References `HepMC::detail::establish_input_stream_info()`, `HepMC::detail::establish_output_stream_info()`, and `HepMC::IO_Exception::InputAndOutput`.

8.23.2.2 HepMC::IO_GenEvent::IO_GenEvent (std::istream &)

constructor requiring an input stream

Definition at line 51 of file IO_GenEvent.cc.

References HepMC::detail::establish_input_stream_info().

8.23.2.3 HepMC::IO_GenEvent::IO_GenEvent (std::ostream &)

constructor requiring an output stream

Definition at line 62 of file IO_GenEvent.cc.

References HepMC::detail::establish_output_stream_info().

8.23.2.4 HepMC::IO_GenEvent::~~IO_GenEvent () [virtual]

Definition at line 73 of file IO_GenEvent.cc.

References HepMC::write_HepMC_IO_block_end().

8.23.3 Member Function Documentation

8.23.3.1 void HepMC::IO_GenEvent::clear () [inline]

clear the IO stream

Definition at line 146 of file IO_GenEvent.h.

8.23.3.2 const std::string & HepMC::IO_GenEvent::error_message () const [inline]

the read error message string

Definition at line 158 of file IO_GenEvent.h.

8.23.3.3 const int HepMC::IO_GenEvent::error_type () const [inline]

integer (enum) associated with read error

Definition at line 154 of file IO_GenEvent.h.

8.23.3.4 bool HepMC::IO_GenEvent::fill_next_event (GenEvent * evt) [virtual]

get the next event

Implements HepMC::IO_BaseClass p. (classHepMC₁₁IO_{BaseClass}_{1df fb95a44d521af510f6431a30f942} ??)

Definition at line 110 of file IO_GenEvent.cc.

References HepMC::GenEvent::clear(), HepMC::IO_Exception::InvalidData, HepMC::GenEvent::is_valid(), HepMC::IO_Exception::NullEvent, HepMC::IO_Exception::OK, and HepMC::IO_Exception::WrongFileType.

8.23.3.5 `bool HepMC::IO_GenEvent::fill_particle_data_table (ParticleDataTable *)` `[inline, virtual]`

`fill_particle_data_table` is required by **IO_BaseClass** (p. 158), but not used here

Implements **HepMC::IO_BaseClass p.** (classHepMC₁₁*IO_BaseClass*_{094fc72dd621a2283989525497715feb} ??)

Definition at line 164 of file `IO_GenEvent.h`.

8.23.3.6 `void HepMC::IO_GenEvent::precision (int)`

set output precision The default precision is 16.

Definition at line 97 of file `IO_GenEvent.cc`.

8.23.3.7 `void HepMC::IO_GenEvent::print (std::ostream & ostr = std::cout) const` `[virtual]`

write to ostr

Reimplemented from **HepMC::IO_BaseClass p.** (classHepMC₁₁*IO_BaseClass*_{8a23f5de9c6bb10931dcacdeb7677413} ??)

Definition at line 87 of file `IO_GenEvent.cc`.

8.23.3.8 `int HepMC::IO_GenEvent::rdstate () const` `[inline]`

check the state of the IO stream

Definition at line 136 of file `IO_GenEvent.h`.

Referenced by `main()`.

8.23.3.9 `bool HepMC::IO_GenEvent::read_io_particle_data_table (std::istream *, ParticleDataTable *)` `[protected]`

Definition at line 179 of file `IO_GenEvent.cc`.

References `read_particle_data()`.

8.23.3.10 `ParticleData * HepMC::IO_GenEvent::read_particle_data (std::istream *, ParticleDataTable *)` `[protected]`

read particle data table information **ParticleDataTable** (p. 211) is deprecated. We include this method for reading old files which may have **ParticleData** (p. 204) information.

Definition at line 187 of file `IO_GenEvent.cc`.

References `HepMC::ParticleDataTable::insert()`.

Referenced by `read_io_particle_data_table()`.

8.23.3.11 void HepMC::IO_GenEvent::use_input_units (Units::MomentumUnit, Units::LengthUnit)

needed when reading a file without units if those units are different than the declared default units (e.g., the default units are MeV, but the file was written with GeV) This method is not necessary if the units are written in the file

Definition at line 80 of file IO_GenEvent.cc.

References HepMC::set_input_units().

8.23.3.12 void HepMC::IO_GenEvent::write_comment (const std::string comment)

insert a comment directly into the output file — normally you only want to do this at the beginning or end of the file. All comments are preceded with "HepMC::IO_GenEvent-COMMENT\n"

Definition at line 163 of file IO_GenEvent.cc.

References HepMC::write_HepMC_IO_block_end(), and HepMC::IO_Exception::WrongFileType.

8.23.3.13 void HepMC::IO_GenEvent::write_event (const GenEvent * evt) [virtual]

write this event

Writes evt to output stream. It does NOT delete the event after writing.

Implements **HepMC::IO_BaseClass p.** (classHepMC₁₁IO_BaseClass_{7929dfd8412207c7904f29652810c1f4} ??)

Definition at line 144 of file IO_GenEvent.cc.

References HepMC::write_HepMC_IO_block_begin(), and HepMC::IO_Exception::WrongFileType.

8.23.3.14 void HepMC::IO_GenEvent::write_particle_data_table (const ParticleDataTable *) [inline, virtual]

write_particle_data_table is required by **IO_BaseClass** (p. 158), but not used here

Implements **HepMC::IO_BaseClass p.** (classHepMC₁₁IO_BaseClass_{1c2fec4084d50bae9ce6b4cbdf133df4} ??)

Definition at line 163 of file IO_GenEvent.h.

The documentation for this class was generated from the following files:

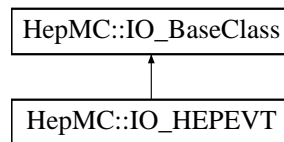
- **IO_GenEvent.h**
- **IO_GenEvent.cc**

8.24 HepMC::IO_HEPEVT Class Reference

HEPEVT IO class.

```
#include <IO_HEPEVT.h>
```

Inheritance diagram for HepMC::IO_HEPEVT::



Public Member Functions

- `IO_HEPEVT ()`
- `virtual ~IO_HEPEVT ()`
- `bool fill_next_event (GenEvent *)`
fill this GenEvent (p. 64)
- `void write_event (const GenEvent *)`
write this GenEvent (p. 64)
- `void print (std::ostream &ostr=std::cout) const`
write output to ostr
- `bool trust_both_mothers_and_daughters () const`
default is false
- `bool trust_mothers_before_daughters () const`
default is true
- `bool print_inconsistency_errors () const`
default is true
- `bool trust_beam_particles () const`
default is true
- `void set_trust_mothers_before_daughters (bool b=true)`
define mother daughter trust rules
- `void set_trust_both_mothers_and_daughters (bool b=false)`
define mother daughter trust rules
- `void set_print_inconsistency_errors (bool b=true)`
- `void set_trust_beam_particles (bool b=true)`
declare whether or not beam particles exist

Protected Member Functions

- **GenParticle * build_particle (int index)**
create a GenParticle (p. 99)
- **void build_production_vertex (int i, std::vector< HepMC::GenParticle * > &hepevt_particle, GenEvent *evt)**
create a production vertex
- **void build_end_vertex (int i, std::vector< HepMC::GenParticle * > &hepevt_particle, GenEvent *evt)**
create an end vertex
- **int find_in_map (const std::map< HepMC::GenParticle *, int > &m, GenParticle *p) const**
find this particle in the particle map

8.24.1 Detailed Description

HEPEVT IO class.

IO class for reading the standard HEPEVT common block.

Examples:

example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc, and testPythiaCopies.cc.

Definition at line 40 of file IO_HEPEVT.h.

8.24.2 Constructor & Destructor Documentation

8.24.2.1 HepMC::IO_HEPEVT::IO_HEPEVT ()

Definition at line 12 of file IO_HEPEVT.cc.

8.24.2.2 HepMC::IO_HEPEVT::~~IO_HEPEVT () [virtual]

Definition at line 18 of file IO_HEPEVT.cc.

8.24.3 Member Function Documentation

8.24.3.1 void HepMC::IO_HEPEVT::build_end_vertex (int i, std::vector< HepMC::GenParticle * > & hepevt_particle, GenEvent * evt) [protected]

create an end vertex

for particle in HEPEVT with index i, build an end vertex if appropriate, and add that vertex to the event

Definition at line 257 of file IO_HEPEVT.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::number_children(), p, HepMC::GenVertex::position(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

8.24.3.2 GenParticle * HepMC::IO_HEPEVT::build_particle (int *index*) [protected]

create a **GenParticle** (p. 99)

Builds a particle object corresponding to index in HEPEVT

Definition at line 325 of file IO_HEPEVT.cc.

References HepMC::HEPEVT_Wrapper::e(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::m(), p, HepMC::HEPEVT_Wrapper::px(), HepMC::HEPEVT_Wrapper::py(), HepMC::HEPEVT_Wrapper::pz(), and HepMC::HEPEVT_Wrapper::status().

Referenced by fill_next_event().

8.24.3.3 void HepMC::IO_HEPEVT::build_production_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create a production vertex

for particle in HEPEVT with index i, build a production vertex if appropriate, and add that vertex to the event

Definition at line 191 of file IO_HEPEVT.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::number_parents(), p, HepMC::GenVertex::position(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

8.24.3.4 bool HepMC::IO_HEPEVT::fill_next_event (GenEvent *) [virtual]

fill this **GenEvent** (p. 64)

Implements HepMC::IO_BaseClass p. (classHepMC₁₁IO_{BaseClass}1dfb95a44d521af510f6431a30f942 ??)

Definition at line 31 of file IO_HEPEVT.cc.

References HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), build_end_vertex(), build_particle(), build_production_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::number_entries(), HepMC::GenEvent::set_beam_particles(), HepMC::GenEvent::set_event_number(), and trust_beam_particles().

8.24.3.5 `int HepMC::IO_HEPEVT::find_in_map (const std::map< HepMC::GenParticle *, int > & m, GenParticle * p) const` [protected]

find this particle in the particle map

Definition at line 340 of file IO_HEPEVT.cc.

References p.

Referenced by write_event().

8.24.3.6 `void HepMC::IO_HEPEVT::print (std::ostream & ostr = std::cout) const` [virtual]

write output to ostr

Reimplemented from **HepMC::IO_BaseClass** p. (classHepMC₁₁*IO_BaseClass*_{8a23f5de9c6bb10931dcacdeb7677413} ??)

Definition at line 20 of file IO_HEPEVT.cc.

8.24.3.7 `bool HepMC::IO_HEPEVT::print_inconsistency_errors () const` [inline]

default is true

Definition at line 126 of file IO_HEPEVT.h.

8.24.3.8 `void HepMC::IO_HEPEVT::set_print_inconsistency_errors (bool b = true)` [inline]

Since HEPEVT has bi-directional pointers, it is possible that the mother/daughter pointers are inconsistent (though physically speaking this should never happen). In practise it happens often. When a conflict occurs (i.e. when mother/daughter pointers are in disagreement, where an empty (0) pointer is not considered a disagreement) an error is printed. These errors can be turned off with: `myio_hepevt.set_print_inconsistency_errors(0)`; but it is **STRONGLY** recommended that you print the HEPEVT common and understand the inconsistency **BEFORE** you turn off the errors. The messages are there for a reason [remember, there is no message printed when the information is missing, ... only when is it inconsistent. User beware.] You can inspect the HEPEVT common block for inconsistencies with **HEPEVT_Wrapper::check_hepevt_consistency()** (p. 144)

There is a switch controlling whether the mother pointers or the daughters are to be trusted. For example, in Pythia the mother information is always correctly included, but the daughter information is often left unfilled: in this case we want to trust the mother pointers and not necessarily the daughters. [THIS IS THE DEFAULT]. Unfortunately the reverse happens for the stdhep(2001) translation of Isajet, so we need an option to toggle the choices.

Definition at line 135 of file IO_HEPEVT.h.

8.24.3.9 `void HepMC::IO_HEPEVT::set_trust_beam_particles (bool b = true)` [inline]

declare whether or not beam particles exist

Definition at line 141 of file IO_HEPEVT.h.

8.24.3.10 `void HepMC::IO_HEPEVT::set_trust_both_mothers_and_daughters (bool b = false)`
`[inline]`

define mother daughter trust rules

Definition at line 129 of file IO_HEPEVT.h.

8.24.3.11 `void HepMC::IO_HEPEVT::set_trust_mothers_before_daughters (bool b = true)`
`[inline]`

define mother daughter trust rules

Definition at line 132 of file IO_HEPEVT.h.

8.24.3.12 `bool HepMC::IO_HEPEVT::trust_beam_particles () const` `[inline]`

default is true

Definition at line 138 of file IO_HEPEVT.h.

Referenced by `fill_next_event()`.

8.24.3.13 `bool HepMC::IO_HEPEVT::trust_both_mothers_and_daughters () const` `[inline]`

default is false

Definition at line 120 of file IO_HEPEVT.h.

8.24.3.14 `bool HepMC::IO_HEPEVT::trust_mothers_before_daughters () const` `[inline]`

default is true

Definition at line 123 of file IO_HEPEVT.h.

8.24.3.15 `void HepMC::IO_HEPEVT::write_event (const GenEvent *)` `[virtual]`

write this **GenEvent** (p. 64)

Implements **HepMC::IO_BaseClass** **p.** (classHepMC11IO_BaseClass7929dfd8412207c7904f29652810c1f4 ??)

Definition at line 110 of file IO_HEPEVT.cc.

References `HepMC::FourVector::e()`, `HepMC::GenEvent::event_number()`, `find_in_map()`, `HepMC::HEPEVT_Wrapper::max_number_entries()`, `p`, `HepMC::FourVector::px()`, `HepMC::FourVector::py()`, `HepMC::FourVector::pz()`, `HepMC::HEPEVT_Wrapper::set_children()`, `HepMC::HEPEVT_Wrapper::set_event_number()`, `HepMC::HEPEVT_Wrapper::set_id()`, `HepMC::HEPEVT_Wrapper::set_mass()`, `HepMC::HEPEVT_Wrapper::set_momentum()`, `HepMC::HEPEVT_Wrapper::set_number_entries()`, `HepMC::HEPEVT_Wrapper::set_parents()`, `HepMC::HEPEVT_Wrapper::set_position()`, `HepMC::HEPEVT_Wrapper::set_status()`, `v`, `HepMC::GenEvent::vertices_begin()`, and `HepMC::GenEvent::vertices_end()`.

The documentation for this class was generated from the following files:

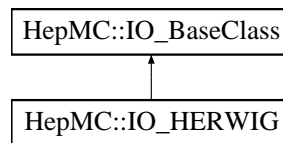
- **IO_HEPEVT.h**
- **IO_HEPEVT.cc**

8.25 HepMC::IO_HERWIG Class Reference

IO_HERWIG (p. 174) is used to get Herwig information.

```
#include <IO_HERWIG.h>
```

Inheritance diagram for HepMC::IO_HERWIG::



Public Member Functions

- **IO_HERWIG ()**
- **virtual ~IO_HERWIG ()**
- **bool fill_next_event (GenEvent *)**
get the next event
- **void print (std::ostream &ostr=std::cout) const**
write to ostr
- **double interfaces_to_version_number () const**
this information is dubious
- **bool print_inconsistency_errors () const**
default is true
- **void set_print_inconsistency_errors (bool b=true)**
decide whether or not to print inconsistency errors
- **bool no_gaps_in_barcodes () const**
ask how to deal with extra non-physical pseudo particles
- **void set_no_gaps_in_barcodes (bool a)**

Protected Member Functions

- **bool trust_both_mothers_and_daughters () const**
default is true
- **bool trust_mothers_before_daughters () const**
default is false
- **void set_trust_mothers_before_daughters (bool b=true)**
define mother daughter trust rules
- **void set_trust_both_mothers_and_daughters (bool b=false)**

define mother daughter trust rules

- **GenParticle * build_particle (int index)**
make a particle
- **void build_production_vertex (int i, std::vector< GenParticle * > &hepevt_particle, GenEvent *evt)**
make a production vertex
- **void build_end_vertex (int i, std::vector< GenParticle * > &hepevt_particle, GenEvent *evt)**
make a decay vertex
- **int find_in_map (const std::map< GenParticle *, int > &m, GenParticle *p) const**
find this particle in the map
- **void repair_hepevt () const**
make the HERWIG HEPEVT common block look like the standard
- **void remove_gaps_in_hepevt () const**
deal with artifacts of repairing HEPEVT
- **void zero_hepevt_entry (int i) const**
zero out a HEPEVT pseudo particle
- **int translate_herwig_to_pdg_id (int i) const**
translate particle ID

8.25.1 Detailed Description

IO_HERWIG (p. 174) is used to get Herwig information.

IO class for reading the HEPEVT common block from the Herwig monte carlo program.

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 57 of file IO_HERWIG.h.

8.25.2 Constructor & Destructor Documentation

8.25.2.1 HepMC::IO_HERWIG::IO_HERWIG ()

Definition at line 12 of file IO_HERWIG.cc.

8.25.2.2 HepMC::IO_HERWIG::~~IO_HERWIG () [virtual]

Definition at line 83 of file IO_HERWIG.cc.

8.25.3 Member Function Documentation

8.25.3.1 `void HepMC::IO_HERWIG::build_end_vertex (int i, std::vector< GenParticle * > & hepevt_particle, GenEvent * evt)` [protected]

make a decay vertex

for particle in HEPEVT with index i, build an end vertex if appropriate, and add that vertex to the event

Definition at line 304 of file IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::number_children(), p, HepMC::GenVertex::position(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

8.25.3.2 `GenParticle * HepMC::IO_HERWIG::build_particle (int index)` [protected]

make a particle

Builds a particle object corresponding to index in HEPEVT

Definition at line 372 of file IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::e(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::m(), p, HepMC::HEPEVT_Wrapper::px(), HepMC::HEPEVT_Wrapper::py(), HepMC::HEPEVT_Wrapper::pz(), and HepMC::HEPEVT_Wrapper::status().

Referenced by fill_next_event().

8.25.3.3 `void HepMC::IO_HERWIG::build_production_vertex (int i, std::vector< GenParticle * > & hepevt_particle, GenEvent * evt)` [protected]

make a production vertex

for particle in HEPEVT with index i, build a production vertex if appropriate, and add that vertex to the event

Definition at line 231 of file IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::number_parents(), p, HepMC::GenVertex::position(), HepMC::GenVertex::print(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

8.25.3.4 `bool HepMC::IO_HERWIG::fill_next_event (GenEvent *)` [virtual]

get the next event

read one event from the Herwig HEPEVT common block and fill **GenEvent** (p.64) return T/F =success/failure

sufficient to do one or the other.

Implements **HepMC::IO_BaseClass** p. (classHepMC₁₁*IO_BaseClass*_{1dfbf95a44d521af510f6431a30f942} ??)

Definition at line 96 of file IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), build_end_vertex(), build_particle(), build_production_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::number_entries(), repair_hepevt(), HepMC::GenEvent::set_beam_particles(), HepMC::GenEvent::set_event_number(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::HEPEVT_Wrapper::status().

8.25.3.5 int HepMC::IO_HERWIG::find_in_map (const std::map< GenParticle *, int > & m, GenParticle * p) const [protected]

find this particle in the map

Definition at line 387 of file IO_HERWIG.cc.

References p.

8.25.3.6 double HepMC::IO_HERWIG::interfaces_to_version_number () const [inline]

this information is dubious

Definition at line 66 of file IO_HERWIG.h.

8.25.3.7 bool HepMC::IO_HERWIG::no_gaps_in_barcodes () const [inline]

ask how to deal with extra non-physical pseudo particles

Definition at line 75 of file IO_HERWIG.h.

8.25.3.8 void HepMC::IO_HERWIG::print (std::ostream & ostr = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** p. (classHepMC₁₁*IO_BaseClass*_{a23f5de9c6bb10931dcacdeb7677413} ??)

Definition at line 85 of file IO_HERWIG.cc.

8.25.3.9 bool HepMC::IO_HERWIG::print_inconsistency_errors () const [inline]

default is true

Definition at line 149 of file IO_HERWIG.h.

8.25.3.10 void HepMC::IO_HERWIG::remove_gaps_in_hepevt () const [protected]

deal with artifacts of repairing HEPEVT

in this scenario, we do not allow there to be zero-ed entries in the HEPEVT common block, and so be reshuffle the common block, removing the zero-ed entries as we go and making sure we keep the mother/daughter relationships appropriate

Definition at line 682 of file IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::e(), HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::m(), HepMC::HEPEVT_Wrapper::number_entries(), HepMC::HEPEVT_Wrapper::px(), HepMC::HEPEVT_Wrapper::py(), HepMC::HEPEVT_Wrapper::pz(), HepMC::HEPEVT_Wrapper::set_children(), HepMC::HEPEVT_Wrapper::set_id(), HepMC::HEPEVT_Wrapper::set_mass(), HepMC::HEPEVT_Wrapper::set_momentum(), HepMC::HEPEVT_Wrapper::set_number_entries(), HepMC::HEPEVT_Wrapper::set_parents(), HepMC::HEPEVT_Wrapper::set_position(), HepMC::HEPEVT_Wrapper::set_status(), HepMC::HEPEVT_Wrapper::status(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by repair_hepevt().

8.25.3.11 void HepMC::IO_HERWIG::repair_hepevt () const [protected]

make the HERWIG HEPEVT common block look like the standard

This routine takes the HEPEVT common block as used in HERWIG, and converts it into the HEPEVT common block in the standard format

This means it:

- removes the color structure, which herwig overloads into the mother/daughter fields
- zeros extra entries for hard subprocess, etc.

Special HERWIG status codes 101,102 colliding beam particles 103 beam-beam collision CMS vector 120 hard subprocess CMS vector 121,122 hard subprocess colliding partons 123-129 hard subprocess outgoing particles 141-149 (ID=94) mirror image of hard subprocess particles 100 (ID=0) cone)

Special HERWIG particle id's 91 clusters 94 jets 0 others with no pdg code

Definition at line 394 of file IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::number_entries(), remove_gaps_in_hepevt(), HepMC::HEPEVT_Wrapper::set_children(), HepMC::HEPEVT_Wrapper::set_id(), HepMC::HEPEVT_Wrapper::set_parents(), HepMC::HEPEVT_Wrapper::status(), translate_herwig_to_pdg_id(), and zero_hepevt_entry().

Referenced by fill_next_event().

8.25.3.12 void HepMC::IO_HERWIG::set_no_gaps_in_barcodes (bool a) [inline]

The HERWIG HEPEVT common block has some EXTRA non-physical ENTRIES (such as CMS frame, HARD subprocess, and CONE). These are removed by **IO_HERWIG** (p. 174). Thus the **HepMC** (p. 19) event will **APPEAR** to have fewer particles in it that herwig did. There is a switch `m_no_gaps_in_barcodes`. For true - then the extra particles are removed from HEPEVT, with the result that the **HepMC** (p. 19) barcodes will be sequential, with no gaps. false - the barcodes will correspond directly to the HEPEVT index, but there will be gaps ... ie some barcodes will be unassigned. this switch requested by I Hinchliffe, October 31, 2002

Definition at line 88 of file IO_HERWIG.h.

8.25.3.13 `void HepMC::IO_HERWIG::set_print_inconsistency_errors (bool b = true)`
[inline]

decide whether or not to print inconsistency errors

Definition at line 158 of file IO_HERWIG.h.

8.25.3.14 `void HepMC::IO_HERWIG::set_trust_both_mothers_and_daughters (bool b = false)`
[inline, protected]

define mother daughter trust rules

Definition at line 152 of file IO_HERWIG.h.

8.25.3.15 `void HepMC::IO_HERWIG::set_trust_mothers_before_daughters (bool b = true)`
[inline, protected]

define mother daughter trust rules

Definition at line 155 of file IO_HERWIG.h.

8.25.3.16 `int HepMC::IO_HERWIG::translate_herwig_to_pdg_id (int i) const` [protected]

translate particle ID

This routine is copied from Lynn Garren's stdhep 5.01. see
<http://cepa.fnal.gov/psm/stdhep/>

Definition at line 753 of file IO_HERWIG.cc.

Referenced by `repair_hepevt()`.

8.25.3.17 `bool HepMC::IO_HERWIG::trust_both_mothers_and_daughters () const` [inline, protected]

default is true

Definition at line 143 of file IO_HERWIG.h.

8.25.3.18 `bool HepMC::IO_HERWIG::trust_mothers_before_daughters () const` [inline, protected]

default is false

Definition at line 146 of file IO_HERWIG.h.

8.25.3.19 `void HepMC::IO_HERWIG::zero_hepevt_entry (int i) const` [protected]

zero out a HEPEVT pseudo particle

Definition at line 742 of file IO_HERWIG.cc.

References `HepMC::HEPEVT_Wrapper::max_number_entries()`, `HepMC::HEPEVT_Wrapper::set_children()`, `HepMC::HEPEVT_Wrapper::set_id()`, `HepMC::HEPEVT_Wrapper::set_mass()`, `HepMC::HEPEVT_Wrapper::set_momentum()`, `HepMC::HEPEVT_Wrapper::set_parents()`, `HepMC::HEPEVT_Wrapper::set_position()`, and `HepMC::HEPEVT_Wrapper::set_status()`.

Referenced by `repair_hepevt()`.

The documentation for this class was generated from the following files:

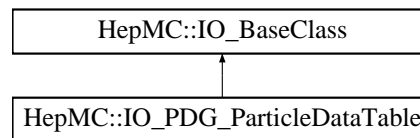
- **IO_HERWIG.h**
- **IO_HERWIG.cc**

8.26 HepMC::IO_PDG_ParticleDataTable Class Reference

an example **ParticleDataTable** (p.211) IO method

```
#include <IO_PDG_ParticleDataTable.h>
```

Inheritance diagram for HepMC::IO_PDG_ParticleDataTable::



Public Member Functions

- **IO_PDG_ParticleDataTable** (const char *filename="PDG98_ParticleDataTable.txt")
constructor using filename
- **virtual ~IO_PDG_ParticleDataTable** ()
- **bool fill_particle_data_table** (ParticleDataTable *)
read the input and fill the table
- **void add_quarks_to_table** (ParticleDataTable &)
add u, d, s, c, b, and t
- **void print** (std::ostream &ostr=std::cout) const
write to ostr
- **int rdstate** () const
check the IO state

Protected Member Functions

- **bool search_for_key_end** (std::istream &in, const char *key)
for internal use
- **void read_entry** (ParticleDataTable *)
read a line

8.26.1 Detailed Description

an example **ParticleDataTable** (p.211) IO method

Example of reading from file PDG98_ParticleDataTable.txt

Definition at line 49 of file IO_PDG_ParticleDataTable.h.

8.26.2 Constructor & Destructor Documentation

8.26.2.1 HepMC::IO_PDG_ParticleDataTable::IO_PDG_ParticleDataTable (const char * *filename* = "PDG98_ParticleDataTable.txt")

constructor using filename

Definition at line 18 of file IO_PDG_ParticleDataTable.cc.

8.26.2.2 HepMC::IO_PDG_ParticleDataTable::~~IO_PDG_ParticleDataTable () [virtual]

Definition at line 26 of file IO_PDG_ParticleDataTable.cc.

8.26.3 Member Function Documentation

8.26.3.1 void HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table (ParticleDataTable &)

add u, d, s, c, b, and t

since quarks aren't included in PDG table, this method adds them

Definition at line 171 of file IO_PDG_ParticleDataTable.cc.

References HepMC::ParticleDataTable::erase(), HepMC::ParticleDataTable::find(), HepMC::ParticleDataTable::insert(), HepMC::ParticleData::mass(), and HepMC::Units::name().

8.26.3.2 bool HepMC::IO_PDG_ParticleDataTable::fill_particle_data_table (ParticleDataTable *) [virtual]

read the input and fill the table

Implements HepMC::IO_BaseClass p. (classHepMC₁₁IO_BaseClass_{094fc72dd621a2283989525497715feb} ??)

Definition at line 30 of file IO_PDG_ParticleDataTable.cc.

References read_entry(), search_for_key_end(), and HepMC::ParticleDataTable::set_description().

8.26.3.3 void HepMC::IO_PDG_ParticleDataTable::print (std::ostream & *ostr* = std::cout) const [inline, virtual]

write to ostr

Reimplemented from HepMC::IO_BaseClass p. (classHepMC₁₁IO_BaseClass_{8a23f5de9c6bb10931dcacdeb7677413} ??)

Definition at line 85 of file IO_PDG_ParticleDataTable.h.

8.26.3.4 int HepMC::IO_PDG_ParticleDataTable::rdstate () const [inline]

check the IO state

Definition at line 63 of file IO_PDG_ParticleDataTable.h.

8.26.3.5 void HepMC::IO_PDG_ParticleDataTable::read_entry (ParticleDataTable *) [protected]

read a line

Definition at line 69 of file IO_PDG_ParticleDataTable.cc.

References HepMC::clifetime_from_width(), HepMC::ParticleDataTable::find(), HepMC::ParticleDataTable::insert(), HepMC::Units::name(), HepMC::ParticleData::set_clifetime(), and HepMC::ParticleData::set_mass().

Referenced by fill_particle_data_table().

8.26.3.6 bool HepMC::IO_PDG_ParticleDataTable::search_for_key_end (std::istream & in, const char * key) [protected]

for internal use

(this method borrowed from IO_Ascii class) reads characters from in until the string of characters matching key is found (success) or EOF is reached (failure). It stops immediately thereafter. Returns T/F for success/fail

Definition at line 209 of file IO_PDG_ParticleDataTable.cc.

Referenced by fill_particle_data_table().

The documentation for this class was generated from the following files:

- IO_PDG_ParticleDataTable.h
- IO_PDG_ParticleDataTable.cc

8.27 HepMC::detail::is_arithmetic< T > Struct Template Reference

undefined and therefore non-arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = false

8.27.1 Detailed Description

```
template<class T> struct HepMC::detail::is_arithmetic< T >
```

undefined and therefore non-arithmetic

Definition at line 22 of file is_arithmetic.h.

8.27.2 Member Data Documentation

8.27.2.1 `template<class T> bool const HepMC::detail::is_arithmetic< T >::value = false`
 [static]

Definition at line 24 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.28 HepMC::detail::is_arithmetic< char > Struct Template Reference

character is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.28.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< char >

character is arithmetic

Definition at line 29 of file is_arithmetic.h.

8.28.2 Member Data Documentation

8.28.2.1 bool const HepMC::detail::is_arithmetic< char >::value = true [static]

Definition at line 30 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.29 HepMC::detail::is_arithmetic< double > Struct Template Reference

double is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = true

8.29.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< double >

double is arithmetic

Definition at line 79 of file is_arithmetic.h.

8.29.2 Member Data Documentation

8.29.2.1 bool const HepMC::detail::is_arithmetic< double >::value = true [static]

Definition at line 80 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.30 HepMC::detail::is_arithmetic< float > Struct Template Reference

float is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.30.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< float >

float is arithmetic

Definition at line 74 of file is_arithmetic.h.

8.30.2 Member Data Documentation

8.30.2.1 bool const HepMC::detail::is_arithmetic< float >::value = true [static]

Definition at line 75 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.31 HepMC::detail::is_arithmetic< int > Struct Template Reference

int is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.31.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< int >

int is arithmetic

Definition at line 54 of file is_arithmetic.h.

8.31.2 Member Data Documentation

8.31.2.1 bool const HepMC::detail::is_arithmetic< int >::value = true [static]

Definition at line 55 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.32 HepMC::detail::is_arithmetic< long > Struct Template Reference

long is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.32.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< long >

long is arithmetic

Definition at line 64 of file is_arithmetic.h.

8.32.2 Member Data Documentation

8.32.2.1 bool const HepMC::detail::is_arithmetic< long >::value = true [static]

Definition at line 65 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.33 HepMC::detail::is_arithmetic< long double > Struct Template Reference

long double is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = true

8.33.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< long double >

long double is arithmetic

Definition at line 84 of file is_arithmetic.h.

8.33.2 Member Data Documentation

8.33.2.1 bool const HepMC::detail::is_arithmetic< long double >::value = true [static]

Definition at line 85 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.34 HepMC::detail::is_arithmetic< short > Struct Template Reference

short is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.34.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< short >

short is arithmetic

Definition at line 44 of file is_arithmetic.h.

8.34.2 Member Data Documentation

8.34.2.1 bool const HepMC::detail::is_arithmetic< short >::value = true [static]

Definition at line 45 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.35 HepMC::detail::is_arithmetic< signed char > Struct Template Reference

signed character is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = true

8.35.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< signed char >

signed character is arithmetic

Definition at line 39 of file is_arithmetic.h.

8.35.2 Member Data Documentation

8.35.2.1 bool const HepMC::detail::is_arithmetic< signed char >::value = true [static]

Definition at line 40 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.36 HepMC::detail::is_arithmetic< unsigned char > Struct Template Reference

unsigned character is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.36.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned char >

unsigned character is arithmetic

Definition at line 34 of file is_arithmetic.h.

8.36.2 Member Data Documentation

8.36.2.1 bool const HepMC::detail::is_arithmetic< unsigned char >::value = true [static]

Definition at line 35 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.37 HepMC::detail::is_arithmetic< unsigned int > Struct Template Reference

unsigned int is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = true

8.37.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned int >

unsigned int is arithmetic

Definition at line 59 of file is_arithmetic.h.

8.37.2 Member Data Documentation

8.37.2.1 bool const HepMC::detail::is_arithmetic< unsigned int >::value = true [static]

Definition at line 60 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.38 HepMC::detail::is_arithmetic< unsigned long > Struct Template Reference

unsigned long is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = **true**

8.38.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned long >

unsigned long is arithmetic

Definition at line 69 of file is_arithmetic.h.

8.38.2 Member Data Documentation

8.38.2.1 bool const HepMC::detail::is_arithmetic< unsigned long >::value = true [static]

Definition at line 70 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.39 HepMC::detail::is_arithmetic< unsigned short > Struct Template Reference

unsigned short is arithmetic

```
#include <is_arithmetic.h>
```

Static Public Attributes

- static bool const **value** = true

8.39.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned short >

unsigned short is arithmetic

Definition at line 49 of file is_arithmetic.h.

8.39.2 Member Data Documentation

8.39.2.1 bool const HepMC::detail::is_arithmetic< unsigned short >::value = true [static]

Definition at line 50 of file is_arithmetic.h.

The documentation for this struct was generated from the following file:

- **is_arithmetic.h**

8.40 IsEventGood Class Reference

example class

Public Member Functions

- **bool operator() (const HepMC::GenEvent *evt)**
check this event for goodness

8.40.1 Detailed Description

example class

event selection predicate. returns true if the event contains a photon with $p_T > 50$ GeV

Examples:

example_EventSelection.cc.

Definition at line 20 of file example_EventSelection.cc.

8.40.2 Member Function Documentation

8.40.2.1 bool IsEventGood::operator() (const HepMC::GenEvent * evt) [inline]

check this event for goodness

Examples:

example_EventSelection.cc.

Definition at line 23 of file example_EventSelection.cc.

References `p`, `HepMC::GenEvent::particles_begin()`, and `HepMC::GenEvent::particles_end()`.

The documentation for this class was generated from the following file:

- **example_EventSelection.cc**

8.41 IsFinalState Class Reference

test class

```
#include <testHepMCIteration.h>
```

Public Member Functions

- **bool operator() (const HepMC::GenParticle *p)**
returns true if the GenParticle does not decay

8.41.1 Detailed Description

test class

this predicate returns true if the input has no decay vertex

Examples:

```
testHepMCIteration.cc.in.
```

Definition at line 26 of file testHepMCIteration.h.

8.41.2 Member Function Documentation

8.41.2.1 bool IsFinalState::operator() (const HepMC::GenParticle *p) [inline]

returns true if the GenParticle does not decay

Definition at line 29 of file testHepMCIteration.h.

References p.

The documentation for this class was generated from the following file:

- **testHepMCIteration.h**

8.42 IsGoodEvent Class Reference

used in the tests

```
#include <IsGoodEvent.h>
```

Public Member Functions

- **bool operator() (const HepMC::GenEvent *evt)**

8.42.1 Detailed Description

used in the tests

event selection predicate. returns true if the event contains a photon with $p_T > 50$ GeV

Examples:

testHepMC.cc.in, testHepMCIteration.cc.in, testMass.cc.in, testMultipleCopies.cc.in, and test-StreamIO.cc.in.

Definition at line 14 of file IsGoodEvent.h.

8.42.2 Member Function Documentation

8.42.2.1 **bool IsGoodEvent::operator() (const HepMC::GenEvent * *evt*)** [inline]

Definition at line 16 of file IsGoodEvent.h.

References `p`, `HepMC::GenEvent::particles_begin()`, and `HepMC::GenEvent::particles_end()`.

The documentation for this class was generated from the following file:

- **IsGoodEvent.h**

8.43 IsGoodEventMyPythia Class Reference

example class

Public Member Functions

- **bool operator() (const HepMC::GenEvent *evt)**
returns true if event is "good"

8.43.1 Detailed Description

example class

event selection predicate. returns true if the event contains a photon with $p_T > 25$ GeV

Examples:

example_MyPythia.cc.

Definition at line 61 of file example_MyPythia.cc.

8.43.2 Member Function Documentation

8.43.2.1 bool IsGoodEventMyPythia::operator() (const HepMC::GenEvent * evt) [inline]

returns true if event is "good"

Examples:

example_MyPythia.cc.

Definition at line 64 of file example_MyPythia.cc.

References `p`, `HepMC::GenEvent::particles_begin()`, and `HepMC::GenEvent::particles_end()`.

The documentation for this class was generated from the following file:

- **example_MyPythia.cc**

8.44 IsPhoton Class Reference

example class

Public Member Functions

- **bool operator() (const HepMC::GenParticle *p)**
returns true if the GenParticle is a photon with more than 10 GeV transverse momentum

8.44.1 Detailed Description

example class

this predicate returns true if the input particle is a photon in the central region ($\eta < 2.5$) with $p_T > 10$ GeV

Examples:

example_UsingIterators.cc.

Definition at line 20 of file example_UsingIterators.cc.

8.44.2 Member Function Documentation

8.44.2.1 bool IsPhoton::operator() (const HepMC::GenParticle *p) [inline]

returns true if the GenParticle is a photon with more than 10 GeV transverse momentum

Examples:

example_UsingIterators.cc.

Definition at line 23 of file example_UsingIterators.cc.

References p.

The documentation for this class was generated from the following file:

- **example_UsingIterators.cc**

8.45 IsStateFinal Class Reference

example class

Public Member Functions

- **bool operator() (const HepMC::GenParticle *p)**
returns true if the GenParticle does not decay

8.45.1 Detailed Description

example class

this predicate returns true if the input has no decay vertex

Examples:

example_UsingIterators.cc.

Definition at line 47 of file example_UsingIterators.cc.

8.45.2 Member Function Documentation

8.45.2.1 bool IsStateFinal::operator() (const HepMC::GenParticle *p) [inline]

returns true if the GenParticle does not decay

Examples:

example_UsingIterators.cc.

Definition at line 50 of file example_UsingIterators.cc.

References p.

The documentation for this class was generated from the following file:

- **example_UsingIterators.cc**

8.46 IsW_Boson Class Reference

example class

Public Member Functions

- **bool operator() (const HepMC::GenParticle *p)**
returns true if the GenParticle is a W

8.46.1 Detailed Description

example class

this predicate returns true if the input particle is a W+/W-

Examples:

example_UsingIterators.cc.

Definition at line 34 of file example_UsingIterators.cc.

8.46.2 Member Function Documentation

8.46.2.1 bool IsW_Boson::operator() (const HepMC::GenParticle *p) [inline]

returns true if the GenParticle is a W

Examples:

example_UsingIterators.cc.

Definition at line 37 of file example_UsingIterators.cc.

References p.

The documentation for this class was generated from the following file:

- **example_UsingIterators.cc**

8.47 HepMC::ParticleData Class Reference

an example **ParticleData** (p. 204) class

```
#include <ParticleData.h>
```

Public Member Functions

- **ParticleData** (std::string name, int id, double charge, double mass=0, double cLifetime=-1, double spin=0)
constructor requiring name, ID, and charge
- **ParticleData** (const char *name, int id, double charge, double mass=0, double cLifetime=-1, double spin=0)
constructor requiring name, ID, and charge
- **virtual ~ParticleData** ()
- **bool operator==** (const ParticleData &) const
equality
- **bool operator!=** (const ParticleData &) const
inequality
- **void print** (std::ostream &ostr=std::cout) const
write particle data information to ostr
- **bool is_lepton** () const
true if charged lepton /neutrino
- **bool is_charged_lepton** () const
true if a charged lepton
- **bool is_em** () const
true if an electron or photon
- **bool is_neutrino** () const
true if a neutrino
- **bool is_hadron** () const
true if a hadron
- **bool is_boson** () const
true if a gauge or higgs boson
- **std::string name** () const
description of the particle according to PDG, i.e. "Delta(1900) S_31"
- **int pdg_id** () const
PDG ID number.

- **double charge () const**
charge
- **double mass () const**
nominal mass
- **double width () const**
width as calculated from clifetime
- **double clifetime () const**
lifetime in mm
- **double spin () const**
J spin.
- **void set_charge (double)**
set charge
- **void set_mass (double)**
set nominal mass
- **void set_width (double)**
set width
- **void set_clifetime (double)**
set lifetime in mm
- **void set_spin (double)**
set J spin

Protected Member Functions

- **int model_independent_pdg_id_ () const**
omits susy/excited/technicolor digit from returned ID

Static Protected Member Functions

- **static unsigned int counter ()**
num ParticleData (p.204) objects in memory

Friends

- **std::ostream & operator<< (std::ostream &, const ParticleData &)**
write to ostr

8.47.1 Detailed Description

an example **ParticleData** (p. 204) class

Particle Data common to all particles of a given PDG id

Examples:

example_BuildEventFromScratch.cc.

Definition at line 69 of file ParticleData.h.

8.47.2 Constructor & Destructor Documentation

8.47.2.1 **HepMC::ParticleData::ParticleData** (std::string *name*, int *id*, double *charge*, double *mass* = 0, double *cLifetime* = -1, double *spin* = 0)

constructor requiring name, ID, and charge

Units (p. 35) ID: defined by PDG group (particles are +ve, antipart are -ve) also consistent with the Pythia definitions charge: fraction of proton charge mass cLifetime: c*time Default mass=0 and cLifetime is -1 which means stable (width= 0.) These defaults exist because many very basic MC generators may produce only massless stable particles in the event record.

Definition at line 12 of file ParticleData.cc.

References set_charge(), and set_spin().

8.47.2.2 **HepMC::ParticleData::ParticleData** (const char * *name*, int *id*, double *charge*, double *mass* = 0, double *cLifetime* = -1, double *spin* = 0)

constructor requiring name, ID, and charge

note, this constructor is redundant to the one above, i.e. one could use: new **HepMC::ParticleData** (p. 204)(string("electron"),11,-1,0.000511,-1,.5); but we keep it because it is convenient.

Definition at line 30 of file ParticleData.cc.

References set_charge(), and set_spin().

8.47.2.3 **HepMC::ParticleData::~ParticleData** () [virtual]

Definition at line 44 of file ParticleData.cc.

8.47.3 Member Function Documentation

8.47.3.1 **double HepMC::ParticleData::charge** () const [inline]

charge

Definition at line 175 of file ParticleData.h.

Referenced by HepMC::operator<<(), and print().

8.47.3.2 double HepMC::ParticleData::clifetime () const [inline]

lifetime in mm

Definition at line 179 of file ParticleData.h.

Referenced by HepMC::operator<<(), and print().

8.47.3.3 unsigned int HepMC::ParticleData::counter () [static, protected]

num **ParticleData** (p. 204) objects in memory

Definition at line 86 of file ParticleData.cc.

8.47.3.4 bool HepMC::ParticleData::is_boson () const [inline]

true if a gauge or higgs boson

true if a gauge or higgs boson -> | 9, 21-39 |

Definition at line 163 of file ParticleData.h.

References pdg_id().

8.47.3.5 bool HepMC::ParticleData::is_charged_lepton () const [inline]

true if a charged lepton

true if a charged lepton -> | 11,13,15 |

Definition at line 146 of file ParticleData.h.

References is_lepton(), and pdg_id().

8.47.3.6 bool HepMC::ParticleData::is_em () const [inline]

true if an electron or photon

true if an electron or photon -> | 11, 22 |

Definition at line 154 of file ParticleData.h.

References pdg_id().

8.47.3.7 bool HepMC::ParticleData::is_hadron () const [inline]

true if a hadron

true if a hadron -> q,g,meson,baryon

Definition at line 158 of file ParticleData.h.

References pdg_id().

8.47.3.8 bool HepMC::ParticleData::is_lepton () const [inline]

true if charged lepton /neutrino

true if a charged lepton or neutrino -> | 11,13,15,12,14,16,17,18 |

Definition at line 142 of file ParticleData.h.

References `pdg_id()`.

Referenced by `is_charged_lepton()`, and `is_neutrino()`.

8.47.3.9 bool HepMC::ParticleData::is_neutrino () const [inline]

true if a neutrino

true if a neutrino -> | 12,14,16 |

Definition at line 150 of file ParticleData.h.

References `is_lepton()`, and `pdg_id()`.

8.47.3.10 double HepMC::ParticleData::mass () const [inline]

nominal mass

Definition at line 178 of file ParticleData.h.

Referenced by `HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table()`, `HepMC::operator<<()`, and `print()`.

8.47.3.11 int HepMC::ParticleData::model_independent_pdg_id_ () const [protected]

omits susy/excited/technicolor digit from returned ID

returns the particle id with the seventh digit removed for susy/excited/technicolor particles. Thus an excited electron (40000011) would be returned as 11 Useful only internally for sorting particles!

Definition at line 61 of file ParticleData.cc.

8.47.3.12 std::string HepMC::ParticleData::name () const [inline]

description of the particle according to PDG, i.e. "Delta(1900) S_31"

Definition at line 173 of file ParticleData.h.

Referenced by `HepMC::operator<<()`, and `print()`.

8.47.3.13 bool HepMC::ParticleData::operator!= (const ParticleData &) const [inline]

inequality

Definition at line 222 of file ParticleData.h.

References `pdg_id()`.

8.47.3.14 bool HepMC::ParticleData::operator== (const ParticleData &) const [inline]

equality

Definition at line 213 of file ParticleData.h.

References m_2spin, m_3charge, m_clifetime, m_mass, and m_pdg_id.

8.47.3.15 int HepMC::ParticleData::pdg_id () const [inline]

PDG ID number.

Definition at line 174 of file ParticleData.h.

Referenced by HepMC::ParticleDataTable::erase(), HepMC::ParticleDataTable::insert(), is_boson(), is_charged_lepton(), is_em(), is_hadron(), is_lepton(), is_neutrino(), operator!(), HepMC::operator<<(), and print().

8.47.3.16 void HepMC::ParticleData::print (std::ostream & ostr = std::cout) const

write particle data information to ostr

Definition at line 48 of file ParticleData.cc.

References charge(), clifetime(), mass(), name(), pdg_id(), and spin().

8.47.3.17 void HepMC::ParticleData::set_charge (double) [inline]

set charge

Definition at line 181 of file ParticleData.h.

Referenced by ParticleData().

8.47.3.18 void HepMC::ParticleData::set_clifetime (double) [inline]

set lifetime in mm

Definition at line 202 of file ParticleData.h.

Referenced by HepMC::IO_PDG_ParticleDataTable::read_entry().

8.47.3.19 void HepMC::ParticleData::set_mass (double) [inline]

set nominal mass

Definition at line 190 of file ParticleData.h.

Referenced by HepMC::IO_PDG_ParticleDataTable::read_entry().

8.47.3.20 void HepMC::ParticleData::set_spin (double) [inline]

set J spin

Definition at line 205 of file ParticleData.h.

Referenced by ParticleData().

8.47.3.21 void HepMC::ParticleData::set_width (double) [inline]

set width

Definition at line 193 of file ParticleData.h.

References HepMC::HepMC_hbarc.

8.47.3.22 double HepMC::ParticleData::spin () const [inline]

J spin.

Definition at line 180 of file ParticleData.h.

Referenced by HepMC::operator<<(), and print().

8.47.3.23 double HepMC::ParticleData::width () const

width as calculated from clifetime

Definition at line 71 of file ParticleData.cc.

References HepMC::HepMC_hbarc.

8.47.4 Friends And Related Function Documentation

8.47.4.1 std::ostream& operator<< (std::ostream & ostr, const ParticleData & pdata) [friend]

write to ostr

Definition at line 94 of file ParticleData.cc.

The documentation for this class was generated from the following files:

- ParticleData.h
- ParticleData.cc

8.48 HepMC::ParticleDataTable Class Reference

an example **ParticleDataTable** (p. 211) class

```
#include <ParticleDataTable.h>
```

Public Types

- **typedef std::map< int, HepMC::ParticleData * >::iterator iterator**
iterator for ParticleData (p. 204) map
- **typedef std::map< int, HepMC::ParticleData * >::const_iterator const_iterator**
const iterator for ParticleData (p. 204) map

Public Member Functions

- **ParticleDataTable (std::string description=std::string())**
constructor with optional description
- **ParticleDataTable (const char description)**
constructor with description
- **ParticleDataTable (const ParticleDataTable &)**
copy constructor
- **virtual ~ParticleDataTable ()**
Shallow: does not delete ParticleData (p. 204) entries.
- **ParticleDataTable & operator= (const ParticleDataTable &)**
shallow: does not copy the entries, only makes new pointers
- **void make_antiparticles_from_particles ()**
make corresponding anti-particles for all particles in table
- **int merge_table (const ParticleDataTable &)**
merge two tables
- **void print (std::ostream &ostr=std::cout) const**
write the table to ostr
- **void delete_all ()**
delete all ParticleData (p. 204) instances in this table
- **void clear ()**
clears table without deleting
- **ParticleData * operator[] (int id) const**
return pointer to requested ParticleData (p. 204)

- **ParticleData * find (int id) const**
return pointer to requested ParticleData (p.204)
- **int size () const**
size of table
- **bool empty () const**
true if the table is empty
- **bool insert (ParticleData *)**
true if successful
- **bool erase (ParticleData *)**
removes from table - does not delete
- **bool erase (int id)**
removes from table - does not delete
- **iterator begin ()**
begin iteration
- **iterator end ()**
end iteration
- **const_iterator begin () const**
begin const iteration
- **const_iterator end () const**
end const iteration
- **std::string description () const**
table description
- **void set_description (std::string)**
set table description
- **void set_description (const char)**
set table description

8.48.1 Detailed Description

an example **ParticleDataTable** (p. 211) class

Example container for **ParticleData** (p. 204) instances. Basically just an interface to STL map.

Examples:

example_BuildEventFromScratch.cc.

Definition at line 35 of file ParticleDataTable.h.

8.48.2 Member Typedef Documentation

8.48.2.1 `typedef std::map<int,HepMC::ParticleData*>::const_iterator HepMC::ParticleDataTable::const_iterator`

const iterator for **ParticleData** (p. 204) map

Definition at line 77 of file ParticleDataTable.h.

8.48.2.2 `typedef std::map<int,HepMC::ParticleData*>::iterator HepMC::ParticleDataTable::iterator`

iterator for **ParticleData** (p. 204) map

Definition at line 75 of file ParticleDataTable.h.

8.48.3 Constructor & Destructor Documentation

8.48.3.1 `HepMC::ParticleDataTable::ParticleDataTable (std::string description = std::string()) [inline]`

constructor with optional description

Definition at line 107 of file ParticleDataTable.h.

8.48.3.2 `HepMC::ParticleDataTable::ParticleDataTable (const char description) [inline]`

constructor with description

Definition at line 114 of file ParticleDataTable.h.

8.48.3.3 `HepMC::ParticleDataTable::ParticleDataTable (const ParticleDataTable &) [inline]`

copy constructor

Definition at line 122 of file ParticleDataTable.h.

8.48.3.4 `HepMC::ParticleDataTable::~~ParticleDataTable () [inline, virtual]`

Shallow: does not delete **ParticleData** (p. 204) entries.

Definition at line 126 of file ParticleDataTable.h.

8.48.4 Member Function Documentation

8.48.4.1 `ParticleDataTable::const_iterator HepMC::ParticleDataTable::begin () const [inline]`

begin const iteration

Definition at line 222 of file ParticleDataTable.h.

8.48.4.2 ParticleDataTable::iterator HepMC::ParticleDataTable::begin () [inline]

begin iteration

Definition at line 214 of file ParticleDataTable.h.

Referenced by make_antiparticles_from_particles(), and merge_table().

8.48.4.3 void HepMC::ParticleDataTable::clear () [inline]

clears table without deleting

Definition at line 249 of file ParticleDataTable.h.

Referenced by delete_all().

8.48.4.4 void HepMC::ParticleDataTable::delete_all () [inline]

delete all **ParticleData** (p. 204) instances in this table

deletes all **ParticleData** (p. 204) instances in this table

Examples:**example_BuildEventFromScratch.cc.**

Definition at line 242 of file ParticleDataTable.h.

References clear().

Referenced by main().

8.48.4.5 std::string HepMC::ParticleDataTable::description () const [inline]

table description

Definition at line 230 of file ParticleDataTable.h.

8.48.4.6 bool HepMC::ParticleDataTable::empty () const [inline]

true if the table is empty

Definition at line 189 of file ParticleDataTable.h.

8.48.4.7 ParticleDataTable::const_iterator HepMC::ParticleDataTable::end () const [inline]

end const iteration

Definition at line 226 of file ParticleDataTable.h.

8.48.4.8 ParticleDataTable::iterator HepMC::ParticleDataTable::end () [inline]

end iteration

Definition at line 218 of file ParticleDataTable.h.

Referenced by make_antiparticles_from_particles(), and merge_table().

8.48.4.9 bool HepMC::ParticleDataTable::erase (int id) [inline]

removes from table - does not delete

removes from table does not delete returns True is an entry pdata existed in the table and was erased

Definition at line 208 of file ParticleDataTable.h.

8.48.4.10 bool HepMC::ParticleDataTable::erase (ParticleData *) [inline]

removes from table - does not delete

removes from table does not delete returns True is an entry pdata existed in the table and was erased

Definition at line 201 of file ParticleDataTable.h.

References HepMC::ParticleData::pdg_id().

Referenced by HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table().

8.48.4.11 ParticleData * HepMC::ParticleDataTable::find (int id) const [inline]

return pointer to requested **ParticleData** (p. 204)

finds a **ParticleData** (p. 204) pointer corresponding to id IF it exists in the table. If not returns NULL

Definition at line 173 of file ParticleDataTable.h.

Referenced by HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table(), operator[](), and HepMC::IO_PDG_ParticleDataTable::read_entry().

8.48.4.12 bool HepMC::ParticleDataTable::insert (ParticleData *) [inline]

true if successful

inserts pdata in the table IFF pdata's id has not already been used. It does NOT replace entries with the same id. True if successful. If you wish to overwrite another entry, first use **erase()** (p. 215)

Examples:

example_BuildEventFromScratch.cc.

Definition at line 193 of file ParticleDataTable.h.

References HepMC::ParticleData::pdg_id().

Referenced by HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table(), main(), make_antiparticles_from_particles(), merge_table(), HepMC::IO_PDG_ParticleDataTable::read_entry(), and HepMC::IO_GenEvent::read_particle_data().

8.48.4.13 void HepMC::ParticleDataTable::make_antiparticles_from_particles () [inline]

make corresponding anti-particles for all particles in table

make corresponding anti-particles for all particles in table

Definition at line 136 of file ParticleDataTable.h.

References begin(), end(), insert(), merge_table(), and p.

8.48.4.14 `int HepMC::ParticleDataTable::merge_table (const ParticleDataTable &) [inline]`

merge two tables

merges pdt into this table each entry from pdt is inserted only if this table does not already have an entry matching the ParticleData's id returns the number of new entries inserted into this table.

Definition at line 251 of file ParticleDataTable.h.

References `begin()`, `end()`, `insert()`, and `p`.

Referenced by `make_antiparticles_from_particles()`.

8.48.4.15 `ParticleDataTable & HepMC::ParticleDataTable::operator= (const ParticleDataTable &) [inline]`

shallow: does not copy the entries, only makes new pointers

Definition at line 128 of file ParticleDataTable.h.

References `m_data_table`, and `m_description`.

8.48.4.16 `ParticleData * HepMC::ParticleDataTable::operator[] (int id) const [inline]`

return pointer to requested **ParticleData** (p. 204)

Definition at line 181 of file ParticleDataTable.h.

References `find()`.

8.48.4.17 `void HepMC::ParticleDataTable::print (std::ostream & ostr = std::cout) const [inline]`

write the table to ostr

prints a summary of all particle Data currently in memory

Examples:

example_BuildEventFromScratch.cc.

Definition at line 153 of file ParticleDataTable.h.

References `size()`.

Referenced by `main()`.

8.48.4.18 `void HepMC::ParticleDataTable::set_description (const char) [inline]`

set table description

Definition at line 238 of file ParticleDataTable.h.

8.48.4.19 `void HepMC::ParticleDataTable::set_description (std::string) [inline]`

set table description

Definition at line 234 of file ParticleDataTable.h.

Referenced by HepMC::IO_PDG_ParticleDataTable::fill_particle_data_table().

8.48.4.20 int HepMC::ParticleDataTable::size () const [inline]

size of table

Definition at line 185 of file ParticleDataTable.h.

Referenced by print().

The documentation for this class was generated from the following file:

- **ParticleDataTable.h**

8.49 HepMC::PdfInfo Class Reference

The **PdfInfo** (p. 218) class stores PDF information.

```
#include <PdfInfo.h>
```

Public Member Functions

- **PdfInfo ()**
default constructor
- **PdfInfo (int i1, int i2, double x1, double x2, double q, double p1, double p2, int pdf_id1=0, int pdf_id2=0)**
all values EXCEPT pdf_id1 and pdf_id2 must be provided
- **~PdfInfo ()**
- **PdfInfo (PdfInfo const &orig)**
copy constructor
- **PdfInfo & operator= (PdfInfo const &rhs)**
make a copy
- **void swap (PdfInfo &other)**
swap two PdfInfo (p. 218) objects
- **bool operator== (const PdfInfo &) const**
check for equality
- **bool operator!= (const PdfInfo &) const**
check for inequality
- **int id1 () const**
flavour code of first parton
- **int id2 () const**
flavour code of second parton
- **int pdf_id1 () const**
LHAPDF set id of first parton.
- **int pdf_id2 () const**
LHAPDF set id of second parton.
- **double x1 () const**
fraction of beam momentum carried by first parton ("beam side")
- **double x2 () const**
fraction of beam momentum carried by second parton ("target side")
- **double scalePDF () const**

Q-scale used in evaluation of PDF's (in GeV).

- **double pdf1 () const**
*PDF (id1, x1, Q) - x*f(x).*
- **double pdf2 () const**
*PDF (id2, x2, Q) - x*f(x).*
- **bool is_valid () const**
verify that the instance contains non-zero information
- **void set_id1 (const int &i)**
set flavour code of first parton
- **void set_id2 (const int &i)**
set flavour code of second parton
- **void set_pdf_id1 (const int &i)**
set LHAPDF set id of first parton
- **void set_pdf_id2 (const int &i)**
set LHAPDF set id of second parton
- **void set_x1 (const double &f)**
set fraction of beam momentum carried by first parton ("beam side")
- **void set_x2 (const double &f)**
set fraction of beam momentum carried by second parton ("target side")
- **void set_scalePDF (const double &f)**
set Q-scale used in evaluation of PDF's (in GeV)
- **void set_pdf1 (const double &f)**
*set x*f(x) of first parton*
- **void set_pdf2 (const double &f)**
*set x*f(x) of second parton*

8.49.1 Detailed Description

The **PdfInfo** (p. 218) class stores PDF information.

HepMC::PdfInfo (p. 218) stores additional PDF information for a **GenEvent** (p. 64). Creation and use of this information is optional.

- int id1; // flavour code of first parton
- int id2; // flavour code of second parton
- int pdf_id1; // LHAPDF set id of first parton (zero by default)

- `int pdf_id2;` // LHAPDF set id of second parton (zero by default)
- `double x1;` // fraction of beam momentum carried by first parton ("beam side")
- `double x2;` // fraction of beam momentum carried by second parton ("target side")
- `double scalePDF;` // Q-scale used in evaluation of PDF's (in GeV)
- `double pdf1;` // PDF (id1, x1, Q)
- `double pdf2;` // PDF (id2, x2, Q)

Input parton flavour codes id1 & id2 are expected to obey the PDG code conventions, especially $g = 21$.

The contents of pdf1 and pdf2 are expected to be $x \cdot f(x)$. The LHAPDF set ids are the entries in the first column of <http://projects.hepforge.org/lhapdf/PDFsets.index>

Examples:

`testMass.cc.in.`

Definition at line 37 of file PdfInfo.h.

8.49.2 Constructor & Destructor Documentation

8.49.2.1 HepMC::PdfInfo::PdfInfo () [inline]

default constructor

Definition at line 43 of file PdfInfo.h.

8.49.2.2 HepMC::PdfInfo::PdfInfo (int i1, int i2, double x1, double x2, double q, double p1, double p2, int pdf_id1 = 0, int pdf_id2 = 0) [inline]

all values EXCEPT pdf_id1 and pdf_id2 must be provided

Definition at line 136 of file PdfInfo.h.

8.49.2.3 HepMC::PdfInfo::~~PdfInfo () [inline]

Definition at line 60 of file PdfInfo.h.

8.49.2.4 HepMC::PdfInfo::PdfInfo (PdfInfo const & orig) [inline]

copy constructor

Definition at line 150 of file PdfInfo.h.

8.49.3 Member Function Documentation

8.49.3.1 int HepMC::PdfInfo::id1 () const [inline]

flavour code of first parton

Definition at line 75 of file PdfInfo.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

8.49.3.2 int HepMC::PdfInfo::id2 () const [inline]

flavour code of second parton

Definition at line 77 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.3 bool HepMC::PdfInfo::is_valid () const [inline]

verify that the instance contains non-zero information

Definition at line 202 of file PdfInfo.h.

Referenced by HepMC::GenEvent::read().

8.49.3.4 bool HepMC::PdfInfo::operator!= (const PdfInfo &) const [inline]

check for inequality

any nonmatching member generates inequality

Definition at line 196 of file PdfInfo.h.

8.49.3.5 PdfInfo & HepMC::PdfInfo::operator= (PdfInfo const & rhs) [inline]

make a copy

Definition at line 162 of file PdfInfo.h.

References swap().

8.49.3.6 bool HepMC::PdfInfo::operator== (const PdfInfo &) const [inline]

check for equality

equality requires that each member match

Definition at line 182 of file PdfInfo.h.

References id1(), id2(), pdf1(), pdf2(), pdf_id1(), pdf_id2(), scalePDF(), x1(), and x2().

8.49.3.7 double HepMC::PdfInfo::pdf1 () const [inline]

PDF (id1, x1, Q) - x*f(x).

Definition at line 89 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.8 double HepMC::PdfInfo::pdf2 () const [inline]

PDF (id2, x2, Q) - x*f(x).

Definition at line 91 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.9 int HepMC::PdfInfo::pdf_id1 () const [inline]

LHAPDF set id of first parton.

Definition at line 79 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.10 int HepMC::PdfInfo::pdf_id2 () const [inline]

LHAPDF set id of second parton.

Definition at line 81 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.11 double HepMC::PdfInfo::scalePDF () const [inline]

Q-scale used in evaluation of PDF's (in GeV).

Definition at line 87 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.12 void HepMC::PdfInfo::set_id1 (const int &i) [inline]

set flavour code of first parton

Definition at line 98 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.13 void HepMC::PdfInfo::set_id2 (const int &i) [inline]

set flavour code of second parton

Definition at line 100 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.14 void HepMC::PdfInfo::set_pdf1 (const double &f) [inline]

set x*f(x) of first parton

Definition at line 112 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.15 void HepMC::PdfInfo::set_pdf2 (const double &f) [inline]

set x*f(x) of second parton

Definition at line 114 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.16 void HepMC::PdfInfo::set_pdf_id1 (const int &i) [inline]

set LHAPDF set id of first parton

Definition at line 102 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.17 void HepMC::PdfInfo::set_pdf_id2 (const int &i) [inline]

set LHAPDF set id of second parton

Definition at line 104 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.18 void HepMC::PdfInfo::set_scalePDF (const double &f) [inline]

set Q-scale used in evaluation of PDF's (in GeV)

Definition at line 110 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.19 void HepMC::PdfInfo::set_x1 (const double &f) [inline]

set fraction of beam momentum carried by first parton ("beam side")

Definition at line 106 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.20 void HepMC::PdfInfo::set_x2 (const double &f) [inline]

set fraction of beam momentum carried by second parton ("target side")

Definition at line 108 of file PdfInfo.h.

Referenced by HepMC::operator>>().

8.49.3.21 void HepMC::PdfInfo::swap (PdfInfo &other) [inline]

swap two **PdfInfo** (p.218) objects

Definition at line 169 of file PdfInfo.h.

References m_id1, m_id2, m_pdf1, m_pdf2, m_pdf_id1, m_pdf_id2, m_scalePDF, m_x1, and m_x2.

Referenced by operator=().

8.49.3.22 double HepMC::PdfInfo::x1 () const [inline]

fraction of beam momentum carried by first parton ("beam side")

Definition at line 83 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

8.49.3.23 double HepMC::PdfInfo::x2 () const [inline]

fraction of beam momentum carried by second parton ("target side")

Definition at line 85 of file PdfInfo.h.

Referenced by `HepMC::operator<<()`, and `operator==()`.

The documentation for this class was generated from the following file:

- PdfInfo.h

8.50 HepMC::Polarization Class Reference

The **Polarization** (p.225) class stores theta and phi for a **GenParticle** (p.99) .

```
#include <Polarization.h>
```

Public Member Functions

- **Polarization (double theta=0, double phi=0)**
default constructor
- **Polarization (const Polarization &inpolar)**
construct from another polarization object
- **Polarization (const ThreeVector &vec3in)**
construct using the polar and azimuthal angles from a ThreeVector (p.238)
- **virtual ~Polarization ()**
- **void swap (Polarization &other)**
swap
- **Polarization & operator= (const Polarization &inpolar)**
make a copy
- **bool operator== (const Polarization &) const**
equality requires that theta and phi are equal
- **bool operator!= (const Polarization &) const**
inequality results if either theta or phi differ
- **void print (std::ostream &ostr=std::cout) const**
print theta and phi
- **double theta () const**
returns polar angle in radians
- **double phi () const**
returns azimuthal angle in radians
- **ThreeVector normal3d () const**
unit 3 vector for easy manipulation
- **double set_theta (double theta)**
set polar angle in radians
- **double set_phi (double phi)**
set azimuthal angle in radians
- **void set_theta_phi (double theta, double phi)**

set both polar and azimuthal angles in radians

- **ThreeVector set_normal3d (const ThreeVector &vec3in)**

sets polarization according to direction of 3 vec

Friends

- `std::ostream & operator<< (std::ostream &, const Polarization &)`

print polarization information

8.50.1 Detailed Description

The **Polarization** (p.225) class stores theta and phi for a **GenParticle** (p.99).

HepMC::Polarization (p.225) stores a particle's theta and phi in radians. Use of this information is optional. By default, the polarization is set to zero.

Definition at line 29 of file Polarization.h.

8.50.2 Constructor & Destructor Documentation

8.50.2.1 HepMC::Polarization::Polarization (double *theta* = 0, double *phi* = 0)

default constructor

Definition at line 11 of file Polarization.cc.

8.50.2.2 HepMC::Polarization::Polarization (const Polarization & *inpolar*)

construct from another polarization object

Definition at line 16 of file Polarization.cc.

8.50.2.3 HepMC::Polarization::Polarization (const ThreeVector & *vec3in*)

construct using the polar and azimuthal angles from a **ThreeVector** (p.238)

Definition at line 21 of file Polarization.cc.

8.50.2.4 virtual HepMC::Polarization::~~Polarization () [inline, virtual]

Definition at line 41 of file Polarization.h.

8.50.3 Member Function Documentation

8.50.3.1 ThreeVector HepMC::Polarization::normal3d () const

unit 3 vector for easy manipulation

Definition at line 47 of file Polarization.cc.

References `phi()`, `HepMC::ThreeVector::setPhi()`, `HepMC::ThreeVector::setTheta()`, and `theta()`.

8.50.3.2 bool HepMC::Polarization::operator!= (const Polarization &) const [inline]

inequality results if either theta or phi differ

Definition at line 98 of file Polarization.h.

8.50.3.3 Polarization & HepMC::Polarization::operator= (const Polarization & *inpolar*)

make a copy

best practices implementation

Definition at line 32 of file Polarization.cc.

References `swap()`.

8.50.3.4 bool HepMC::Polarization::operator== (const Polarization &) const [inline]

equality requires that theta and phi are equal

Definition at line 93 of file Polarization.h.

References `phi()`, and `theta()`.

8.50.3.5 double HepMC::Polarization::phi () const [inline]

returns azimuthal angle in radians

Definition at line 87 of file Polarization.h.

Referenced by `normal3d()`, `HepMC::operator<<()`, and `operator==()`.

8.50.3.6 void HepMC::Polarization::print (std::ostream & *ostr* = std::cout) const

print theta and phi

Definition at line 39 of file Polarization.cc.

8.50.3.7 ThreeVector HepMC::Polarization::set_normal3d (const ThreeVector & *vec3in*)

sets polarization according to direction of 3 vec

Definition at line 72 of file Polarization.cc.

References `HepMC::ThreeVector::phi()`, `set_phi()`, `set_theta()`, and `HepMC::ThreeVector::theta()`.

8.50.3.8 double HepMC::Polarization::set_phi (double *phi*)

set azimuthal angle in radians

Phi is restricted to be between 0 -> 2pi if an out of range value is given, it is translated to this range.

Definition at line 61 of file Polarization.cc.

Referenced by set_normal3d(), and set_theta_phi().

8.50.3.9 double HepMC::Polarization::set_theta (double *theta*)

set polar angle in radians

Theta is restricted to be between 0 -> pi if an out of range value is given, it is translated to this range.

Definition at line 55 of file Polarization.cc.

Referenced by set_normal3d(), and set_theta_phi().

8.50.3.10 void HepMC::Polarization::set_theta_phi (double *theta*, double *phi*)

set both polar and azimuthal angles in radians

Definition at line 67 of file Polarization.cc.

References set_phi(), and set_theta().

8.50.3.11 void HepMC::Polarization::swap (Polarization & *other*)

swap

Definition at line 26 of file Polarization.cc.

References m_phi, and m_theta.

Referenced by operator=(), and HepMC::GenParticle::swap().

8.50.3.12 double HepMC::Polarization::theta () const [inline]

returns polar angle in radians

Definition at line 86 of file Polarization.h.

Referenced by normal3d(), HepMC::operator<<(), and operator==().

8.50.4 Friends And Related Function Documentation**8.50.4.1 std::ostream& operator<< (std::ostream & *ostr*, const Polarization & *polar*) [friend]**

print polarization information

Definition at line 107 of file Polarization.cc.

The documentation for this class was generated from the following files:

- **Polarization.h**
- **Polarization.cc**

8.51 HepMC::StreamInfo Class Reference

StreamInfo (p.230) contains extra information needed when using streaming IO.

```
#include <StreamInfo.h>
```

Public Member Functions

- **StreamInfo ()**
default constructor
- **~StreamInfo ()**
destructor
- **std::string IO_GenEvent_Key () const**
IO_GenEvent (p.164) begin event block key.
- **std::string IO_GenEvent_End () const**
IO_GenEvent (p.164) end event block key.
- **std::string IO_Ascii_Key () const**
- **std::string IO_Ascii_End () const**
IO_Ascii end event block key.
- **std::string IO_Ascii_PDT_Key () const**
IO_Ascii begin particle data block key.
- **std::string IO_Ascii_PDT_End () const**
IO_Ascii end particle data block key.
- **std::string IO_ExtendedAscii_Key () const**
- **std::string IO_ExtendedAscii_End () const**
IO_ExtendedAscii end event block key.
- **std::string IO_ExtendedAscii_PDT_Key () const**
IO_ExtendedAscii begin particle data block key.
- **std::string IO_ExtendedAscii_PDT_End () const**
IO_ExtendedAscii end particle data block key.
- **int io_type () const**
get IO type
- **void set_io_type (int)**
set IO type
- **bool has_key () const**
- **void set_has_key (bool)**
set to false if the stream does not have a file type key

- **Units::MomentumUnit io_momentum_unit () const**
get the I/O momentum units
- **Units::LengthUnit io_position_unit () const**
get the I/O length units
- **int stream_id () const**
- **bool finished_first_event () const**
Special information is processed the first time we use the IO.
- **void set_finished_first_event (bool b)**
Special information is processed the first time we use the IO.
- **void use_input_units (Units::MomentumUnit, Units::LengthUnit)**

8.51.1 Detailed Description

StreamInfo (p.230) contains extra information needed when using streaming IO.

This class contains the extra information needed when using streaming IO to process **HepMC** (p.19) GenEvents

Definition at line 26 of file StreamInfo.h.

8.51.2 Constructor & Destructor Documentation

8.51.2.1 HepMC::StreamInfo::StreamInfo ()

default constructor

Definition at line 13 of file StreamInfo.cc.

8.51.2.2 HepMC::StreamInfo::~~StreamInfo () [inline]

destructor

Definition at line 31 of file StreamInfo.h.

8.51.3 Member Function Documentation

8.51.3.1 bool HepMC::StreamInfo::finished_first_event () const [inline]

Special information is processed the first time we use the IO.

Definition at line 81 of file StreamInfo.h.

Referenced by `HepMC::detail::establish_input_stream_info()`, `HepMC::establish_input_stream_info()`, `HepMC::detail::establish_output_stream_info()`, `HepMC::establish_output_stream_info()`, `HepMC::GenEvent::read()`, `HepMC::GenEvent::write()`, `HepMC::write_HepMC_IO_block_begin()`, and `HepMC::write_HepMC_IO_block_end()`.

8.51.3.2 bool HepMC::StreamInfo::has_key () const [inline]

true if the stream has a file type key has_key is true by default

Definition at line 67 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read().

8.51.3.3 std::string HepMC::StreamInfo::IO_Ascii_End () const [inline]

IO_Ascii end event block key.

Definition at line 43 of file StreamInfo.h.

8.51.3.4 std::string HepMC::StreamInfo::IO_Ascii_Key () const [inline]

IO_Ascii begin event block key IO_Ascii has been removed, but we want to be able to read existing files written by IO_Ascii

Definition at line 41 of file StreamInfo.h.

8.51.3.5 std::string HepMC::StreamInfo::IO_Ascii_PDT_End () const [inline]

IO_Ascii end particle data block key.

Definition at line 47 of file StreamInfo.h.

8.51.3.6 std::string HepMC::StreamInfo::IO_Ascii_PDT_Key () const [inline]

IO_Ascii begin particle data block key.

Definition at line 45 of file StreamInfo.h.

8.51.3.7 std::string HepMC::StreamInfo::IO_ExtendedAscii_End () const [inline]

IO_ExtendedAscii end event block key.

Definition at line 54 of file StreamInfo.h.

8.51.3.8 std::string HepMC::StreamInfo::IO_ExtendedAscii_Key () const [inline]

IO_ExtendedAscii begin event block key IO_ExtendedAscii has been removed, but we want to be able to read existing files written by IO_ExtendedAscii

Definition at line 52 of file StreamInfo.h.

8.51.3.9 std::string HepMC::StreamInfo::IO_ExtendedAscii_PDT_End () const [inline]

IO_ExtendedAscii end particle data block key.

Definition at line 58 of file StreamInfo.h.

8.51.3.10 `std::string HepMC::StreamInfo::IO_ExtendedAscii_PDT_Key () const` [inline]

IO_ExtendedAscii begin particle data block key.

Definition at line 56 of file StreamInfo.h.

8.51.3.11 `std::string HepMC::StreamInfo::IO_GenEvent_End () const` [inline]

IO_GenEvent (p.164) end event block key.

Definition at line 36 of file StreamInfo.h.

Referenced by HepMC::write_HepMC_IO_block_end().

8.51.3.12 `std::string HepMC::StreamInfo::IO_GenEvent_Key () const` [inline]

IO_GenEvent (p.164) begin event block key.

Definition at line 34 of file StreamInfo.h.

Referenced by HepMC::write_HepMC_IO_block_begin().

8.51.3.13 `Units::MomentumUnit HepMC::StreamInfo::io_momentum_unit () const` [inline]

get the I/O momentum units

Definition at line 72 of file StreamInfo.h.

Referenced by HepMC::detail::read_units().

8.51.3.14 `Units::LengthUnit HepMC::StreamInfo::io_position_unit () const` [inline]

get the I/O length units

Definition at line 74 of file StreamInfo.h.

Referenced by HepMC::detail::read_units().

8.51.3.15 `int HepMC::StreamInfo::io_type () const` [inline]

get IO type

Definition at line 61 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read(), and HepMC::detail::read_particle().

8.51.3.16 `void HepMC::StreamInfo::set_finished_first_event (bool b)` [inline]

Special information is processed the first time we use the IO.

Definition at line 83 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read(), and HepMC::GenEvent::write().

8.51.3.17 void HepMC::StreamInfo::set_has_key (bool)

set to false if the stream does not have a file type key

Definition at line 46 of file StreamInfo.cc.

8.51.3.18 void HepMC::StreamInfo::set_io_type (int)

set IO type

Definition at line 42 of file StreamInfo.cc.

8.51.3.19 int HepMC::StreamInfo::stream_id () const [inline]

get the I/O stream id This is used for sanity checking.

Definition at line 78 of file StreamInfo.h.

Referenced by HepMC::HepMCStreamCallback().

8.51.3.20 void HepMC::StreamInfo::use_input_units (Units::MomentumUnit, Units::LengthUnit)

needed when reading a file without units if those units are different than the declared default units (e.g., the default units are MeV, but the file was written with GeV) This method is not necessary if the units are written in the file

Definition at line 37 of file StreamInfo.cc.

Referenced by HepMC::set_input_units().

The documentation for this class was generated from the following files:

- **StreamInfo.h**
- **StreamInfo.cc**

8.52 HepMC::TempParticleMap Class Reference

TempParticleMap (p.235) is a temporary GenParticle* container used during input.

```
#include <TempParticleMap.h>
```

Public Types

- `typedef std::map< HepMC::GenParticle *, int > TempMap`
- `typedef std::map< int, HepMC::GenParticle * > TempOrderMap`
- `typedef TempMap::iterator TempMapIterator`
- `typedef TempOrderMap::iterator orderIterator`

Public Member Functions

- `TempParticleMap ()`
- `~TempParticleMap ()`
- `TempMapIterator begin ()`
- `TempMapIterator end ()`
- `orderIterator order_begin ()`
- `orderIterator order_end ()`
- `int end_vertex (GenParticle *)`
- `void addEndParticle (GenParticle *, int &)`

8.52.1 Detailed Description

TempParticleMap (p.235) is a temporary GenParticle* container used during input.

Used by IO classes for recoverable particle ordering. Map GenParticle* against both outgoing vertex and particle order.

Definition at line 24 of file TempParticleMap.h.

8.52.2 Member Typedef Documentation

8.52.2.1 `typedef TempOrderMap::iterator HepMC::TempParticleMap::orderIterator`

Definition at line 29 of file TempParticleMap.h.

8.52.2.2 `typedef std::map<HepMC::GenParticle*,int> HepMC::TempParticleMap::TempMap`

Definition at line 26 of file TempParticleMap.h.

8.52.2.3 `typedef TempMap::iterator HepMC::TempParticleMap::TempMapIterator`

Definition at line 28 of file TempParticleMap.h.

8.52.2.4 `typedef std::map<int,HepMC::GenParticle*> HepMC::TempParticleMap::TempOrder-Map`

Definition at line 27 of file `TempParticleMap.h`.

8.52.3 Constructor & Destructor Documentation

8.52.3.1 `HepMC::TempParticleMap::TempParticleMap ()` [inline]

Definition at line 31 of file `TempParticleMap.h`.

8.52.3.2 `HepMC::TempParticleMap::~~TempParticleMap ()` [inline]

Definition at line 34 of file `TempParticleMap.h`.

8.52.4 Member Function Documentation

8.52.4.1 `void HepMC::TempParticleMap::addEndParticle (GenParticle *, int &)` [inline]

Definition at line 58 of file `TempParticleMap.h`.

References `p`.

Referenced by `HepMC::detail::read_particle()`.

8.52.4.2 `TempMapIterator HepMC::TempParticleMap::begin ()` [inline]

Definition at line 36 of file `TempParticleMap.h`.

8.52.4.3 `TempMapIterator HepMC::TempParticleMap::end ()` [inline]

Definition at line 37 of file `TempParticleMap.h`.

Referenced by `end_vertex()`.

8.52.4.4 `int HepMC::TempParticleMap::end_vertex (GenParticle *)` [inline]

Definition at line 50 of file `TempParticleMap.h`.

References `end()`, and `p`.

Referenced by `HepMC::GenEvent::read()`.

8.52.4.5 `orderIterator HepMC::TempParticleMap::order_begin ()` [inline]

Definition at line 38 of file `TempParticleMap.h`.

Referenced by `HepMC::GenEvent::read()`.

8.52.4.6 orderIterator HepMC::TempParticleMap::order_end () [inline]

Definition at line 39 of file TempParticleMap.h.

Referenced by HepMC::GenEvent::read().

The documentation for this class was generated from the following file:

- TempParticleMap.h

8.53 HepMC::ThreeVector Class Reference

ThreeVector (p.238) is a simple representation of a position or displacement 3 vector.

```
#include <SimpleVector.h>
```

Public Member Functions

- **ThreeVector (double xin, double yin=0, double zin=0)**
construct using x, y, and z (only x is required)
- **ThreeVector ()**
- **template<class T> ThreeVector (const T &v, typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type !=0)**
- **ThreeVector (const ThreeVector &v)**
copy constructor
- **void swap (ThreeVector &other)**
swap
- **double x () const**
return x
- **double y () const**
return y
- **double z () const**
return z
- **void setX (double x)**
set x
- **void setY (double y)**
set y
- **void setZ (double z)**
set z
- **void set (double x, double y, double z)**
set x, y, and z
- **double phi () const**
The azimuth angle.
- **double theta () const**
The polar angle.
- **double r () const**
The magnitude.

- **void setPhi (double)**
Set phi keeping magnitude and theta constant (BaBar).
- **void setTheta (double)**
Set theta keeping magnitude and phi constant (BaBar).
- **double perp2 () const**
The transverse component squared (ρ^2 in cylindrical coordinate system).
- **double perp () const**
The transverse component (ρ in cylindrical coordinate system).
- **ThreeVector & operator= (const ThreeVector &)**
make a copy
- **bool operator== (const ThreeVector &) const**
equality
- **bool operator!= (const ThreeVector &) const**
inequality

8.53.1 Detailed Description

ThreeVector (p.238) is a simple representation of a position or displacement 3 vector.

For compatibility with existing code, the basic expected geometrical access methods are provided. Also, there is a templated constructor that will take another vector (HepLorentzVector, GenVector, ...) which must have the following methods: **x()** (p.243), **y()** (p.243), **z()** (p.243).

Examples:

testSimpleVector.cc, and VectorConversion.h.

Definition at line 131 of file SimpleVector.h.

8.53.2 Constructor & Destructor Documentation

8.53.2.1 HepMC::ThreeVector::ThreeVector (double *xin*, double *yin* = 0, double *zin* = 0) [inline]

construct using x, y, and z (only x is required)

Definition at line 136 of file SimpleVector.h.

8.53.2.2 HepMC::ThreeVector::ThreeVector () [inline]

Definition at line 139 of file SimpleVector.h.

8.53.2.3 `template<class T> HepMC::ThreeVector::ThreeVector (const T & v, typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type * = 0) [inline]`

templated constructor this is used ONLY if T is not arithmetic

Definition at line 145 of file SimpleVector.h.

8.53.2.4 `HepMC::ThreeVector::ThreeVector (const ThreeVector & v) [inline]`

copy constructor

Definition at line 150 of file SimpleVector.h.

8.53.3 Member Function Documentation

8.53.3.1 `bool HepMC::ThreeVector::operator!= (const ThreeVector &) const`

inequality

8.53.3.2 `ThreeVector& HepMC::ThreeVector::operator= (const ThreeVector &)`

make a copy

8.53.3.3 `bool HepMC::ThreeVector::operator== (const ThreeVector &) const`

equality

8.53.3.4 `double HepMC::ThreeVector::perp () const`

The transverse component (rho in cylindrical coordinate system).

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.53.3.5 `double HepMC::ThreeVector::perp2 () const`

The transverse component squared (rho^2 in cylindrical coordinate system).

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.53.3.6 double HepMC::ThreeVector::phi () const

The azimuth angle.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`, and `HepMC::Polarization::set_normal3d()`.

8.53.3.7 double HepMC::ThreeVector::r () const

The magnitude.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.53.3.8 void HepMC::ThreeVector::set (double x, double y, double z)

set *x*, *y*, and *z*

Examples:

`testSimpleVector.cc.`

Referenced by `main()`.

8.53.3.9 void HepMC::ThreeVector::setPhi (double)

Set phi keeping magnitude and theta constant (BaBar).

Examples:

`testSimpleVector.cc.`

Referenced by `main()`, and `HepMC::Polarization::normal3d()`.

8.53.3.10 void HepMC::ThreeVector::setTheta (double)

Set theta keeping magnitude and phi constant (BaBar).

Examples:

`testSimpleVector.cc.`

Referenced by `main()`, and `HepMC::Polarization::normal3d()`.

8.53.3.11 void HepMC::ThreeVector::setX (double *x*) [inline]

set *x*

Examples:

`testSimpleVector.cc.`

Definition at line 159 of file SimpleVector.h.

Referenced by `main()`.

8.53.3.12 void HepMC::ThreeVector::setY (double *y*) [inline]

set *y*

Examples:

`testSimpleVector.cc.`

Definition at line 160 of file SimpleVector.h.

Referenced by `main()`.

8.53.3.13 void HepMC::ThreeVector::setZ (double *z*) [inline]

set *z*

Examples:

`testSimpleVector.cc.`

Definition at line 161 of file SimpleVector.h.

Referenced by `main()`.

8.53.3.14 void HepMC::ThreeVector::swap (ThreeVector & *other*)

swap

8.53.3.15 double HepMC::ThreeVector::theta () const

The polar angle.

Examples:

`testSimpleVector.cc.`

Referenced by `main()`, and `HepMC::Polarization::set_normal3d()`.

8.53.3.16 double HepMC::ThreeVector::x () const [inline]

return x

Examples:

testSimpleVector.cc.

Definition at line 155 of file SimpleVector.h.

Referenced by main().

8.53.3.17 double HepMC::ThreeVector::y () const [inline]

return y

Examples:

testSimpleVector.cc.

Definition at line 156 of file SimpleVector.h.

Referenced by main().

8.53.3.18 double HepMC::ThreeVector::z () const [inline]

return z

Examples:

testSimpleVector.cc.

Definition at line 157 of file SimpleVector.h.

Referenced by main().

The documentation for this class was generated from the following file:

- **SimpleVector.h**

8.54 HepMC::WeightContainer Class Reference

Container for the Weights associated with an event or vertex.

```
#include <WeightContainer.h>
```

Public Types

- `typedef std::vector< double >::iterator iterator`
iterator for the weight container
- `typedef std::vector< double >::const_iterator const_iterator`
const iterator for the weight container

Public Member Functions

- `WeightContainer (unsigned int n=0, const double &value=0.)`
default constructor
- `WeightContainer (const std::vector< double > &weights)`
construct from a vector of weights
- `WeightContainer (const WeightContainer &in)`
copy
- `virtual ~WeightContainer ()`
- `void swap (WeightContainer &other)`
swap
- `WeightContainer & operator= (const WeightContainer &)`
copy
- `WeightContainer & operator= (const std::vector< double > &in)`
copy
- `void print (std::ostream &ostr=std::cout) const`
print weights
- `int size () const`
size of weight container
- `bool empty () const`
return true if weight container is empty
- `void push_back (const double &)`
push onto weight container
- `void pop_back ()`
pop from weight container

- **void clear ()**
clear the weight container
- **double & operator[] (unsigned int n)**
access the weight container
- **const double & operator[] (unsigned int n) const**
access the weight container
- **double & front ()**
returns the first element
- **const double & front () const**
returns the first element
- **double & back ()**
returns the last element
- **const double & back () const**
returns the last element
- **iterator begin ()**
begining of the weight container
- **iterator end ()**
end of the weight container
- **const_iterator begin () const**
begining of the weight container
- **const_iterator end () const**
end of the weight container

8.54.1 Detailed Description

Container for the Weights associated with an event or vertex.
Basically just an interface to STL vector.
Definition at line 24 of file WeightContainer.h.

8.54.2 Member Typedef Documentation

8.54.2.1 `typedef std::vector<double>::const_iterator HepMC::WeightContainer::const_iterator`

const iterator for the weight container
Definition at line 73 of file WeightContainer.h.

8.54.2.2 `typedef std::vector<double>::iterator HepMC::WeightContainer::iterator`

iterator for the weight container

Definition at line 71 of file `WeightContainer.h`.

8.54.3 Constructor & Destructor Documentation

8.54.3.1 `HepMC::WeightContainer::WeightContainer (unsigned int n = 0, const double & value = 0.) [inline]`

default constructor

Definition at line 91 of file `WeightContainer.h`.

8.54.3.2 `HepMC::WeightContainer::WeightContainer (const std::vector< double > & weights) [inline]`

construct from a vector of weights

Definition at line 96 of file `WeightContainer.h`.

8.54.3.3 `HepMC::WeightContainer::WeightContainer (const WeightContainer & in) [inline]`

copy

Definition at line 100 of file `WeightContainer.h`.

8.54.3.4 `HepMC::WeightContainer::~~WeightContainer () [inline, virtual]`

Definition at line 104 of file `WeightContainer.h`.

8.54.4 Member Function Documentation

8.54.4.1 `const double & HepMC::WeightContainer::back () const [inline]`

returns the last element

Definition at line 158 of file `WeightContainer.h`.

8.54.4.2 `double & HepMC::WeightContainer::back () [inline]`

returns the last element

Definition at line 156 of file `WeightContainer.h`.

8.54.4.3 `WeightContainer::const_iterator HepMC::WeightContainer::begin () const [inline]`

begining of the weight container

Definition at line 167 of file `WeightContainer.h`.

8.54.4.4 WeightContainer::iterator HepMC::WeightContainer::begin () [inline]

beginning of the weight container

Definition at line 161 of file WeightContainer.h.

Referenced by `print()`, and `HepMC::IO_AsciiParticles::write_event()`.

8.54.4.5 void HepMC::WeightContainer::clear () [inline]

clear the weight container

Definition at line 143 of file WeightContainer.h.

8.54.4.6 bool HepMC::WeightContainer::empty () const [inline]

return true if weight container is empty

Definition at line 136 of file WeightContainer.h.

8.54.4.7 WeightContainer::const_iterator HepMC::WeightContainer::end () const [inline]

end of the weight container

Definition at line 170 of file WeightContainer.h.

8.54.4.8 WeightContainer::iterator HepMC::WeightContainer::end () [inline]

end of the weight container

Definition at line 164 of file WeightContainer.h.

Referenced by `print()`, `HepMC::GenVertex::print()`, `HepMC::GenEvent::print()`, `HepMC::GenEvent::write()`, and `HepMC::IO_AsciiParticles::write_event()`.

8.54.4.9 const double & HepMC::WeightContainer::front () const [inline]

returns the first element

Definition at line 153 of file WeightContainer.h.

8.54.4.10 double & HepMC::WeightContainer::front () [inline]

returns the first element

Definition at line 151 of file WeightContainer.h.

8.54.4.11 WeightContainer & HepMC::WeightContainer::operator= (const std::vector< double > & in) [inline]

copy

best practices implementation

Definition at line 118 of file WeightContainer.h.

8.54.4.12 **WeightContainer & HepMC::WeightContainer::operator= (const WeightContainer &)** [inline]

copy

best practices implementation

Definition at line 110 of file WeightContainer.h.

8.54.4.13 **const double & HepMC::WeightContainer::operator[] (unsigned int *n*) const** [inline]

access the weight container

Definition at line 148 of file WeightContainer.h.

8.54.4.14 **double & HepMC::WeightContainer::operator[] (unsigned int *n*)** [inline]

access the weight container

Definition at line 145 of file WeightContainer.h.

8.54.4.15 **void HepMC::WeightContainer::pop_back ()** [inline]

pop from weight container

Definition at line 141 of file WeightContainer.h.

8.54.4.16 **void HepMC::WeightContainer::print (std::ostream & *ostr* = std::cout) const** [inline]

print weights

Definition at line 125 of file WeightContainer.h.

References `begin()`, and `end()`.

8.54.4.17 **void HepMC::WeightContainer::push_back (const double &)** [inline]

push onto weight container

Definition at line 138 of file WeightContainer.h.

8.54.4.18 **int HepMC::WeightContainer::size () const** [inline]

size of weight container

Definition at line 134 of file WeightContainer.h.

Referenced by `HepMC::compareWeights()`, `HepMC::GenVertex::print()`, `HepMC::GenEvent::print()`, `HepMC::GenEvent::write()`, and `HepMC::IO_AsciiParticles::write_event()`.

8.54.4.19 void HepMC::WeightContainer::swap (WeightContainer & *other*) [inline]

swap

Definition at line 106 of file WeightContainer.h.

References `m_weights`.

Referenced by `HepMC::GenVertex::swap()`, and `HepMC::GenEvent::swap()`.

The documentation for this class was generated from the following file:

- **WeightContainer.h**

Chapter 9

HepMC File Documentation

9.1 CompareGenEvent.cc File Reference

```
#include <iostream>
#include "HepMC/CompareGenEvent.h"
#include "HepMC/GenEvent.h"
```

Namespaces

- namespace **HepMC**

Functions

- `bool HepMC::compareGenEvent (GenEvent *, GenEvent *)`
- `bool HepMC::compareSignalProcessVertex (GenEvent *, GenEvent *)`
- `bool HepMC::compareBeamParticles (GenEvent *, GenEvent *)`
- `bool HepMC::compareWeights (GenEvent *, GenEvent *)`
- `bool HepMC::compareParticles (GenEvent *, GenEvent *)`
- `bool HepMC::compareVertices (GenEvent *, GenEvent *)`
- `bool HepMC::compareVertex (GenVertex *v1, GenVertex *v2)`

9.2 CompareGenEvent.h File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Namespaces

- namespace **HepMC**

Functions

- `bool HepMC::compareGenEvent (GenEvent *, GenEvent *)`
- `bool HepMC::compareSignalProcessVertex (GenEvent *, GenEvent *)`
- `bool HepMC::compareBeamParticles (GenEvent *, GenEvent *)`
- `bool HepMC::compareWeights (GenEvent *, GenEvent *)`
- `bool HepMC::compareVertices (GenEvent *, GenEvent *)`
- `bool HepMC::compareParticles (GenEvent *, GenEvent *)`
- `bool HepMC::compareVertex (GenVertex *v1, GenVertex *v2)`

9.3 enable_if.h File Reference

Namespaces

- namespace **HepMC**
- namespace **HepMC::detail**

Classes

- struct **HepMC::detail::enable_if**<, >
internal - used to decide if a class is arithmetic
- struct **HepMC::detail::enable_if**< true, T >
internal - use if class T is arithmetic
- struct **HepMC::detail::disable_if**<, >
internal - used by SimpleVector to decide if a class is arithmetic
- struct **HepMC::detail::disable_if**< false, T >
internal - used by SimpleVector to decide if a class is arithmetic

9.4 example_BuildEventFromScratch.cc File Reference

```
#include <iostream>
#include "VectorConversion.h"
#include "HepMC/GenEvent.h"
#include "HepMC/ParticleDataTable.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Functions

- `int main()`

9.4.1 Function Documentation

9.4.1.1 `int main()`

Examples:

`example_BuildEventFromScratch.cc`, `example_EventSelection.cc`, `example_MyHerwig.cc`, `example_MyPythia.cc`, `example_MyPythiaOnlyToHepMC.cc`, `example_PythiaStreamIO.cc`, `example_UsingIterators.cc`, `testFlow.cc`, `testHepMC.cc.in`, `testHepMCIteration.cc.in`, `testHerwigCopies.cc`, `testMass.cc.in`, `testMultipleCopies.cc.in`, `testPrintBug.cc`, `testPythiaCopies.cc`, `testSimpleVector.cc`, `testStreamIO.cc.in`, and `testUnits.cc`.

Definition at line 26 of file `example_BuildEventFromScratch.cc`.

References `HepMC::GenVertex::add_particle_in()`, `HepMC::GenVertex::add_particle_out()`, `HepMC::GenEvent::add_vertex()`, `HepMC::lifetime_from_width()`, `convertTo()`, `HepMC::ParticleDataTable::delete_all()`, `HepMC::Units::GEV`, `HepMC::ParticleDataTable::insert()`, `HepMC::Units::MM`, `p`, `HepMC::GenEvent::particles_begin()`, `HepMC::GenEvent::particles_end()`, `HepMC::GenEvent::print()`, `HepMC::ParticleDataTable::print()`, `HepMC::GenEvent::set_signal_process_vertex()`, and `HepMC::GenEvent::use_units()`.

9.5 example_EventSelection.cc File Reference

```
#include "HepMC/IO_GenEvent.h"  
#include "HepMC/GenEvent.h"
```

Classes

- class **IsEventGood**
example class

Functions

- int **main()**

9.5.1 Function Documentation

9.5.1.1 int main()

Definition at line 37 of file example_EventSelection.cc.

References `HepMC::GenEvent::event_number()`, and `HepMC::IO_BaseClass::read_next_event()`.

9.6 example_MyHerwig.cc File Reference

```
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
#include "HerwigHelper.h"
```

Functions

- `int main()`

9.6.1 Function Documentation

9.6.1.1 `int main()`

To Compile: go to the **HepMC** (p.19) directory and type: `gmake examples/example_MyHerwig.exe`

In this example the precision and number of entries for the HEPEVT fortran common block are explicitly defined to correspond to those used in the Herwig version of the HEPEVT common block. If you get funny output from HEPEVT in your own code, probably you have set these values incorrectly!

Definition at line 25 of file `example_MyHerwig.cc`.

References `getHerwigCrossSection()`, `HepMC::Units::GEV`, `hwbgen`, `hwbmch`, `hwcdec`, `hwcfor`, `hwdhad`, `hwdhob`, `hwdhvy`, `hwefin`, `hweini`, `hwepro`, `hwevnt`, `hwigin`, `hwmevt`, `hwproc`, `hwufne`, `hwuinc`, `hwuine`, `HepMC::Units::MM`, `HepMC::GenEvent::print()`, `HepMC::HEPEVT_Wrapper::print_hepevt()`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::GenEvent::set_event_number()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_signal_process_id()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, and `HepMC::GenEvent::use_units()`.

9.7 example_MyPythia.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Classes

- class **IsGoodEventMyPythia**
example class

Functions

- void **pythia_out** ()
- void **pythia_in** ()
- void **pythia_in_out** ()
- void **event_selection** ()
- void **pythia_particle_out** ()
- int **main** ()

9.7.1 Function Documentation

9.7.1.1 void event_selection ()

Examples:

example_MyPythia.cc.

Definition at line 152 of file example_MyPythia.cc.

References `getPythiaCrossSection()`, `HepMC::Units::GEV`, `initPythia()`, `HepMC::Units::MM`, `pypars`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_mpi()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, and `HepMC::GenEvent::use_units()`.

Referenced by `main()`.

9.7.1.2 int main ()

Definition at line 85 of file example_MyPythia.cc.

References `event_selection()`, `pythia_in()`, `pythia_in_out()`, and `pythia_out()`.

9.7.1.3 void pythia_in ()

Examples:

example_MyPythia.cc.

Definition at line 205 of file example_MyPythia.cc.

References HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

Referenced by main().

9.7.1.4 void pythia_in_out ()

Examples:

example_MyPythia.cc.

Definition at line 239 of file example_MyPythia.cc.

References HepMC::GenEvent::event_number(), getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

Referenced by main().

9.7.1.5 void pythia_out ()

Examples:

example_MyPythia.cc.

Definition at line 99 of file example_MyPythia.cc.

References getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

Referenced by main().

9.7.1.6 void pythia_particle_out ()

Examples:

example_MyPythia.cc.

Definition at line 312 of file example_MyPythia.cc.

References `getPythiaCrossSection()`, `HepMC::Units::GEV`, `initPythia()`, `HepMC::Units::MM`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::GenEvent::set_event_number()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_signal_process_id()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, and `HepMC::GenEvent::use_units()`.

9.8 example_MyPythiaOnlyToHepMC.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- `int main()`

9.8.1 Function Documentation

9.8.1.1 `int main()`

Definition at line 23 of file `example_MyPythiaOnlyToHepMC.cc`.

References `getPythiaCrossSection()`, `HepMC::Units::GEV`, `initPythia()`, `HepMC::Units::MM`, `pypars`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_mpi()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, and `HepMC::GenEvent::use_units()`.

9.9 example_PythiaStreamIO.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- void **writePythiaStreamIO** ()
- void **readPythiaStreamIO** ()
- int **main** ()

9.9.1 Function Documentation

9.9.1.1 int main ()

Definition at line 31 of file example_PythiaStreamIO.cc.

References **readPythiaStreamIO()**, and **writePythiaStreamIO()**.

9.9.1.2 void readPythiaStreamIO ()

Examples:

example_PythiaStreamIO.cc.

Definition at line 103 of file example_PythiaStreamIO.cc.

References **HepMC::GenEvent::cross_section()**, **HepMC::GenEvent::is_valid()**, **HepMC::GenEvent::read()**, **HepMC::GenEvent::write()**, **HepMC::write_HepMC_IO_block_begin()**, and **HepMC::write_HepMC_IO_block_end()**.

Referenced by **main()**.

9.9.1.3 void writePythiaStreamIO ()

example of generating events with Pythia using **HepMC/PythiaWrapper.h** (p.327) Events are read into the **HepMC** (p.19) event record from the FORTRAN HEPEVT common block using the **IO_HEPEVT** strategy

To Compile: go to the **HepMC** (p.19) example directory and type: make **example_PythiaStreamIO.exe**

This example uses streaming I/O **writePythiaStreamIO()** (p.261) sets the cross section in GenRun **readPythiaStreamIO()** (p.261) reads the file written by **writePythiaStreamIO()** (p.261)

Examples:**example_PythiaStreamIO.cc.**

Definition at line 40 of file example_PythiaStreamIO.cc.

References `getPythiaCrossSection()`, `HepMC::Units::GEV`, `initPythia()`, `HepMC::Units::MM`, `pypars`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::GenEvent::set_event_number()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_mpi()`, `HepMC::GenEvent::set_signal_process_id()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, `HepMC::GenEvent::use_units()`, `HepMC::write_HepMC_IO_block_begin()`, and `HepMC::write_HepMC_IO_block_end()`.

Referenced by `main()`.

9.10 example_UsingIterators.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include <math.h>
#include <algorithm>
#include <list>
```

Classes

- `class IsPhoton`
example class
- `class IsW_Boson`
example class
- `class IsStateFinal`
example class

Functions

- `int main()`

9.10.1 Function Documentation

9.10.1.1 `int main()`

Definition at line 56 of file `example_UsingIterators.cc`.

References `HepMC::copy_if()`, `HepMC::descendants`, `p`, `HepMC::parents`, `HepMC::GenEvent::particles_begin()`, `HepMC::GenEvent::particles_end()`, `HepMC::IO_GenEvent::rdstate()`, `HepMC::IO_BaseClass::read_next_event()`, `v`, `HepMC::GenEvent::vertices_begin()`, and `HepMC::GenEvent::vertices_end()`.

9.11 Flow.cc File Reference

```
#include "HepMC/Flow.h"  
#include "HepMC/GenParticle.h"  
#include "HepMC/GenVertex.h"  
#include "HepMC/SearchVector.h"
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &ostr, const Flow &f)`
for printing

9.12 Flow.h File Reference

```
#include <iostream>
#include <map>
#include <vector>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::Flow**
The flow object.

9.13 GenCrossSection.cc File Reference

```
#include <iostream>
#include <sstream>
#include "HepMC/GenCrossSection.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- namespace **HepMC**

9.14 GenCrossSection.h File Reference

```
#include <iostream>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::GenCrossSection**

*The **GenCrossSection** (p. 60) class stores the generated cross section.*

Functions

- `std::ostream & HepMC::operator<< (std::ostream &os, GenCrossSection &xs)`
- `std::istream & HepMC::operator>> (std::istream &is, GenCrossSection &xs)`

9.15 GenEvent.cc File Reference

```
#include <iomanip>
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/Version.h"
#include "HepMC/StreamHelpers.h"
```

Namespaces

- namespace **HepMC**

9.16 GenEvent.h File Reference

```
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include "HepMC/WeightContainer.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/HeavyIon.h"
#include "HepMC/PdfInfo.h"
#include "HepMC/Units.h"
#include "HepMC/HepMCDefs.h"
#include <map>
#include <string>
#include <vector>
#include <algorithm>
#include <iostream>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::GenEvent**
The GenEvent (p. 64) class is the core of HepMC (p. 19).
- class **HepMC::GenEvent::vertex_const_iterator**
const vertex iterator
- class **HepMC::GenEvent::vertex_iterator**
non-const vertex iterator
- class **HepMC::GenEvent::particle_const_iterator**
const particle iterator
- class **HepMC::GenEvent::particle_iterator**
non-const particle iterator

Functions

- `template<class InputIterator, class OutputIterator, class Predicate> void HepMC::copy_if (InputIterator first, InputIterator last, OutputIterator out, Predicate pred)`
define the type of iterator to use

- **std::ostream & HepMC::operator<< (std::ostream &, GenEvent &)**
standard streaming IO output operator
- **std::istream & HepMC::operator>> (std::istream &, GenEvent &)**
standard streaming IO input operator
- **std::istream & HepMC::set_input_units (std::istream &, Units::MomentumUnit, Units::LengthUnit)**
set the units for this input stream
- **std::ostream & HepMC::write_HepMC_IO_block_begin (std::ostream &)**
Explicitly write the begin block lines that IO_GenEvent (p. 164) uses.
- **std::ostream & HepMC::write_HepMC_IO_block_end (std::ostream &)**
Explicitly write the end block line that IO_GenEvent (p. 164) uses.
- **GenEvent & HepMC::convert_units (GenEvent &evt, Units::MomentumUnit m, Units::LengthUnit l)**

9.17 GenEventStreamIO.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/StreamInfo.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/Version.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- namespace **HepMC**
- namespace **HepMC::detail**

Functions

- **void HepMC::HepMCStreamCallback (std::ios_base::event e, std::ios_base &b, int i)**
- **template<class IO> StreamInfo & HepMC::get_stream_info (IO &iost)**
- **std::ostream & HepMC::operator<< (std::ostream &, GenEvent &)**
standard streaming IO output operator
- **std::istream & HepMC::operator>> (std::istream &, GenEvent &)**
standard streaming IO input operator
- **std::istream & HepMC::set_input_units (std::istream &, Units::MomentumUnit, Units::LengthUnit)**
set the units for this input stream
- **std::ostream & HepMC::write_HepMC_IO_block_begin (std::ostream &)**
Explicitly write the begin block lines that IO_GenEvent (p. 164) uses.
- **std::ostream & HepMC::write_HepMC_IO_block_end (std::ostream &)**
Explicitly write the end block line that IO_GenEvent (p. 164) uses.
- **std::ostream & HepMC::establish_output_stream_info (std::ostream &os)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & HepMC::establish_input_stream_info (std::istream &is)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & HepMC::detail::read_particle (std::istream &, TempParticleMap &, GenParticle *)**

- **std::istream & HepMC::detail::read_units (std::istream &is, GenEvent &evt)**
- **std::ostream & HepMC::detail::establish_output_stream_info (std::ostream &)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & HepMC::detail::establish_input_stream_info (std::istream &)**
used by IO_GenEvent (p. 164) constructor

9.18 GenParticle.cc File Reference

```
#include "HepMC/GenEvent.h"  
#include "HepMC/GenVertex.h"  
#include "HepMC/GenParticle.h"  
#include <iomanip>
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &ostr, const GenParticle &part)`
print particle

9.19 GenParticle.h File Reference

```
#include "HepMC/Flow.h"
#include "HepMC/Polarization.h"
#include "HepMC/SimpleVector.h"
#include <iostream>
#include <stdint.h>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::GenParticle**
The GenParticle (p. 99) class contains information about generated particles.

Defines

- #define **hepmc_uint64_t** uint64_t

9.19.1 Define Documentation

9.19.1.1 #define **hepmc_uint64_t** uint64_t

Definition at line 37 of file GenParticle.h.

9.20 GenVertex.cc File Reference

```
#include "HepMC/GenParticle.h"
#include "HepMC/GenVertex.h"
#include "HepMC/GenEvent.h"
#include "HepMC/SearchVector.h"
#include <iomanip>
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &ostr, const GenVertex &vtx)`
print vertex information

9.21 GenVertex.h File Reference

```
#include "HepMC/WeightContainer.h"
#include "HepMC/SimpleVector.h"
#include <iostream>
#include <iterator>
#include <vector>
#include <set>
#include <algorithm>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::GenVertex**
GenVertex (p. 109) contains information about decay vertices.
- class **HepMC::GenVertex::edge_iterator**
edge iterator
- class **HepMC::GenVertex::vertex_iterator**
vertex iterator
- class **HepMC::GenVertex::particle_iterator**
particle iterator

Enumerations

- enum **HepMC::IteratorRange** {
 HepMC::parents, **HepMC::children**, **HepMC::family**, **HepMC::ancestors**,
 HepMC::descendants, **HepMC::relatives** }
type of iteration

9.22 HeavyIon.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/HeavyIon.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &, HeavyIon const *)`
Write the contents of HeavyIon (p. 133) to an output stream.
- `std::istream & HepMC::operator>> (std::istream &, HeavyIon *)`
Read the contents of HeavyIon (p. 133) from an input stream.

9.23 HeavyIon.h File Reference

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::HeavyIon**
The HeavyIon (p. 133) class stores information about heavy ions.

Functions

- `std::ostream & HepMC::operator<< (std::ostream &, HeavyIon const *)`
Write the contents of HeavyIon (p. 133) to an output stream.
- `std::istream & HepMC::operator>> (std::istream &, HeavyIon *)`
Read the contents of HeavyIon (p. 133) from an input stream.

9.24 HEPEVT_Wrapper.cc File Reference

```
#include "HepMC/HEPEVT_Wrapper.h"
```

Namespaces

- namespace **HepMC**

9.25 HEPEVT_Wrapper.h File Reference

```
#include <ctype.h>
#include <iostream>
#include <cstdio>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::HEPEVT_Wrapper**
Generic Wrapper for the fortran HEPEVT common block.

Defines

- #define **HEPEVT_EntriesAllocation 10000**
- #define **hepevt hepevt_**

Variables

- const unsigned int **hepevt_bytes_allocation**
- struct {
 char data [hepevt_bytes_allocation]
} **hepevt_**

9.25.1 Define Documentation

9.25.1.1 #define hepevt hepevt_

Definition at line 84 of file HEPEVT_Wrapper.h.

Referenced by `HepMC::HEPEVT_Wrapper::byte_num_to_double()`, `HepMC::HEPEVT_Wrapper::byte_num_to_int()`, and `HepMC::HEPEVT_Wrapper::write_byte_num()`.

9.25.1.2 #define HEPEVT_EntriesAllocation 10000

Definition at line 4 of file HEPEVT_Wrapper.h.

9.25.2 Variable Documentation

9.25.2.1 char data[hepevt_bytes_allocation]

Definition at line 81 of file HEPEVT_Wrapper.h.

9.25.2.2 struct { ... } hepevt_

9.25.2.3 const unsigned int hepevt_bytes_allocation

Initial value:

```
sizeof(long int) * ( 2 + 6 * HEPEVT_EntriesAllocation )  
+ sizeof(double) * ( 9 * HEPEVT_EntriesAllocation )
```

Definition at line 66 of file HEPEVT_Wrapper.h.

Referenced by HepMC::HEPEVT_Wrapper::byte_num_to_double(), HepMC::HEPEVT_Wrapper::byte_num_to_int(), and HepMC::HEPEVT_Wrapper::write_byte_num().

9.26 HepMCDefs.h File Reference

Defines

- `#define HEPMC_VERSION "2.05.01"`

9.26.1 Define Documentation

9.26.1.1 `#define HEPMC_VERSION "2.05.01"`

Definition at line 50 of file HepMCDefs.h.

Referenced by `HepMC::versionName()`.

9.27 HerwigHelper.h File Reference

```
#include "HepMC/HerwigWrapper.h"
#include "HepMC/GenCrossSection.h"
```

Functions

- **HepMC::GenCrossSection getHerwigCrossSection (int ngen)**
calculate the Herwig cross section and statistical error

9.27.1 Function Documentation

9.27.1.1 HepMC::GenCrossSection getHerwigCrossSection (int ngen) [inline]

calculate the Herwig cross section and statistical error

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 13 of file HerwigHelper.h.

References hwevnt, and xsec.

Referenced by main().

9.28 HerwigWrapper.h File Reference

```
#include "HepMC/HerwigWrapper6_4.h"
```

9.29 HerwigWrapper6_4.h File Reference

```
#include <ctype.h>
```

Defines

- `#define hwproc hwproc_`
- `#define hwbmch hwbmch_`
- `#define hwevnt hwevnt_`
- `#define hwpram hwpram_`
- `#define hwigin hwigin_`
- `#define hwigup hwigup_`
- `#define hwuinc hwuinc_`
- `#define hwusta hwusta_`
- `#define hweini hweini_`
- `#define hwuine hwuine_`
- `#define hwepro hwepro_`
- `#define hwupro hwupro_`
- `#define hwbgen hwbgen_`
- `#define hwdhob hwdhob_`
- `#define hwcfor hwcfor_`
- `#define hwcdec hwcdec_`
- `#define hwdhad hwdhad_`
- `#define hwdhvy hwdhvy_`
- `#define hwmevt hwmevt_`
- `#define hwufne hwufne_`
- `#define hwefin hwefin_`
- `#define hwudpr hwudpr_`
- `#define hwuepr hwuepr_`
- `#define hwupup hwupup_`
- `#define hwegup hwegup_`
- `#define hwudat hwudat_`

Variables

- ```
struct {
 double EBEAM1
 double EBEAM2
 double PBEAM1
 double PBEAM2
 int IPROC
 int MAXEV
} hwproc_
```
- ```
struct {
    char PART1 [8]
    char PART2 [8]
} hwbmch_
```
- `const int herwig_hepevt_size = 4000`

- struct {
 - double AVWGT
 - double EVWGT
 - double GAMWT
 - double TLOUT
 - double WBIGST
 - double WGTMAX
 - double WGTSUM
 - double WSQSUM
 - int IDHW [herwig_hepevt_size]
 - int IERROR
 - int ISTAT
 - int LWEVT
 - int MAXER
 - int MAXPR
 - int NOWGT
 - int NRN [2]
 - int NUMER
 - int NUMERU
 - int NWGTS
 - int GENSOFF
- } hwevnt_
- struct {
 - double AFCH [2][16]
 - double ALPHEM
 - double B1LIM
 - double BETAF
 - double BTCLM
 - double CAFAC
 - double CFFAC
 - double CLMAX
 - double CLPOW
 - double CLSMR [2]
 - double CSPEED
 - double ENSOFF
 - double ETAMIX
 - double F0MIX
 - double F1MIX
 - double F2MIX
 - double GAMH
 - double GAMW
 - double GAMZ
 - double GAMZP
 - double GEV2NB
 - double H1MIX
 - double PDIQK
 - double PGSMX
 - double PGSPL [4]
 - double PHIMIX
 - double PIFAC
 - double PRSOFF
 - double PSPLT [2]
 - double PTRMS

```
double PXRMS
double QCDL3
double QCDL5
double QCDLAM
double QDIQK
double QFCH [16]
double QG
double QSPAC
double QV
double SCABI
double SWEIN
double TMTOP
double VFCH [2][16]
double VCKM [3][3]
double VGCUT
double VQCUT
double VPCUT
double ZBINM
double EFFMIN
double OMHMIX
double ET2MIX
double PH3MIX
double GCUTME
int IOPREM
int IPRINT
int ISPAC
int LRSUD
int LWSUD
int MODPDF [2]
int NBTRY
int NCOLO
int NCTRY
int NDTRY
int NETRY
int NFLAV
int NGSPL
int NSTRU
int NSTRY
int NZBIN
int IOP4JT [2]
int NPRFMT
int AZSOFT
int AZSPIN
int CLDIR [2]
int HARDME
int NOSPAC
int PRNDEC
int PRVTX
int SOFTME
int ZPRIME
int PRNDEF
int PRNTEX
int PRNWEB
} hwpram_
```

9.29.1 Define Documentation

9.29.1.1 #define hwbgen hwbgen_

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 87 of file HerwigWrapper6_4.h.

Referenced by main().

9.29.1.2 #define hwbmch hwbmch_

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 32 of file HerwigWrapper6_4.h.

Referenced by main().

9.29.1.3 #define hwcdec hwcdec_

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 90 of file HerwigWrapper6_4.h.

Referenced by main().

9.29.1.4 #define hwcfor hwcfor_

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 89 of file HerwigWrapper6_4.h.

Referenced by main().

9.29.1.5 #define hwdhad hwdhad_

Examples:

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 91 of file HerwigWrapper6_4.h.

Referenced by main().

9.29.1.6 #define hwdhob hwdhob_

Examples:

`example_MyHerwig.cc`, and `testHerwigCopies.cc`.

Definition at line 88 of file `HerwigWrapper6_4.h`.

Referenced by `main()`.

9.29.1.7 #define hwdhvy hwdhvy_

Examples:

`example_MyHerwig.cc`, and `testHerwigCopies.cc`.

Definition at line 92 of file `HerwigWrapper6_4.h`.

Referenced by `main()`.

9.29.1.8 #define hwefin hwefin_

Examples:

`example_MyHerwig.cc`, and `testHerwigCopies.cc`.

Definition at line 95 of file `HerwigWrapper6_4.h`.

Referenced by `main()`.

9.29.1.9 #define hwegup hwegup_

Definition at line 100 of file `HerwigWrapper6_4.h`.

9.29.1.10 #define hweini hweini_

Examples:

`example_MyHerwig.cc`, and `testHerwigCopies.cc`.

Definition at line 83 of file `HerwigWrapper6_4.h`.

Referenced by `main()`.

9.29.1.11 #define hwepro hwepro_

Examples:

`example_MyHerwig.cc`, and `testHerwigCopies.cc`.

Definition at line 85 of file `HerwigWrapper6_4.h`.

Referenced by `main()`.

9.29.1.12 #define hwevnt hwevnt_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 46 of file HerwigWrapper6_4.h.

Referenced by `getHerwigCrossSection()`, and `main()`.

9.29.1.13 #define hwigin hwigin_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 79 of file HerwigWrapper6_4.h.

Referenced by `main()`.

9.29.1.14 #define hwigup hwigup_

Definition at line 80 of file HerwigWrapper6_4.h.

9.29.1.15 #define hwmevt hwmevt_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 93 of file HerwigWrapper6_4.h.

Referenced by `main()`.

9.29.1.16 #define hwpram hwpram_

Definition at line 74 of file HerwigWrapper6_4.h.

9.29.1.17 #define hwproc hwproc_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 23 of file HerwigWrapper6_4.h.

Referenced by `main()`.

9.29.1.18 #define hwudat hwudat_**9.29.1.19 #define hwudpr hwudpr_**

Definition at line 97 of file HerwigWrapper6_4.h.

9.29.1.20 #define hwuepr hwuepr_

Definition at line 98 of file HerwigWrapper6_4.h.

9.29.1.21 #define hwufne hwufne_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 94 of file HerwigWrapper6_4.h.

Referenced by `main()`.

9.29.1.22 #define hwuinc hwuinc_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 81 of file HerwigWrapper6_4.h.

Referenced by `main()`.

9.29.1.23 #define hwuine hwuine_**Examples:**

example_MyHerwig.cc, and testHerwigCopies.cc.

Definition at line 84 of file HerwigWrapper6_4.h.

Referenced by `main()`.

9.29.1.24 #define hwupro hwupro_

Definition at line 86 of file HerwigWrapper6_4.h.

9.29.1.25 #define hwupup hwupup_

Definition at line 99 of file HerwigWrapper6_4.h.

9.29.1.26 #define hwusta hwusta_

Definition at line 82 of file HerwigWrapper6_4.h.

9.29.2 Variable Documentation**9.29.2.1 double AFCH[2][16]**

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.2 double ALPHEM

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.3 double AVWGT

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.4 int AZSOFT

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.5 int AZSPIN

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.6 double B1LIM

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.7 double BETAF

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.8 double BTCLM

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.9 double CAFAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.10 double CFFAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.11 int CLDIR[2]

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.12 double CLMAX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.13 double CLPOW

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.14 double CLSMR[2]

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.15 double CSPEED

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.16 double EBEAM1

Definition at line 19 of file HerwigWrapper6_4.h.

9.29.2.17 double EBEAM2

Definition at line 19 of file HerwigWrapper6_4.h.

9.29.2.18 double EFFMIN

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.19 double ENSOF

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.20 double ET2MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.21 double ETAMIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.22 double EVWGT

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.23 double F0MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.24 double F1MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.25 double F2MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.26 double GAMH

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.27 double GAMW

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.28 double GAMWT

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.29 double GAMZ

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.30 double GAMZP

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.31 double GCUTME

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.32 int GENSOFF

Definition at line 43 of file HerwigWrapper6_4.h.

9.29.2.33 double GEV2NB

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.34 double H1MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.35 int HARDME

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.36 const int herwig_hepevt_size = 4000

Definition at line 37 of file HerwigWrapper6_4.h.

9.29.2.37 struct { ... } hwbmch_

9.29.2.38 struct { ... } hwevnt_

9.29.2.39 struct { ... } hwpram_

9.29.2.40 struct { ... } hwproc_

9.29.2.41 int IDHW[herwig_hepevt_size]

Definition at line 41 of file HerwigWrapper6_4.h.

9.29.2.42 int IERROR

Definition at line 41 of file HerwigWrapper6_4.h.

9.29.2.43 int IOP4JT[2]

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.44 int IOPREM

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.45 int IPRINT

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.46 int IPROC

Definition at line 20 of file HerwigWrapper6_4.h.

9.29.2.47 int ISPAC

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.48 int ISTAT

Definition at line 41 of file HerwigWrapper6_4.h.

9.29.2.49 int LRSUD

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.50 int LWEVT

Definition at line 41 of file HerwigWrapper6_4.h.

9.29.2.51 int LWSUD

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.52 int MAXER

Definition at line 41 of file HerwigWrapper6_4.h.

9.29.2.53 int MAXEV

Definition at line 20 of file HerwigWrapper6_4.h.

9.29.2.54 int MAXPR

Definition at line 41 of file HerwigWrapper6_4.h.

9.29.2.55 int MODPDF[2]

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.56 int NBTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.57 int NCOLO

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.58 int NCTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.59 int NDTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.60 int NETRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.61 int NFLAV

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.62 int NGSPL

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.63 int NOSPAC

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.64 int NOWGT

Definition at line 42 of file HerwigWrapper6_4.h.

9.29.2.65 int NPRFMT

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.66 int NRN[2]

Definition at line 42 of file HerwigWrapper6_4.h.

9.29.2.67 int NSTRU

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.68 int NSTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.69 int NUMER

Definition at line 42 of file HerwigWrapper6_4.h.

9.29.2.70 int NUMERU

Definition at line 42 of file HerwigWrapper6_4.h.

9.29.2.71 int NWGTS

Definition at line 42 of file HerwigWrapper6_4.h.

9.29.2.72 int NZBIN

Definition at line 68 of file HerwigWrapper6_4.h.

9.29.2.73 double OMHMIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.74 char PART1[8]

Definition at line 29 of file HerwigWrapper6_4.h.

9.29.2.75 char PART2[8]

Definition at line 29 of file HerwigWrapper6_4.h.

9.29.2.76 double PBEAM1

Definition at line 19 of file HerwigWrapper6_4.h.

9.29.2.77 double PBEAM2

Definition at line 19 of file HerwigWrapper6_4.h.

9.29.2.78 double PDIQK

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.79 double PGSMX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.80 double PGSPL[4]

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.81 double PH3MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.82 double PHIMIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.83 double PIFAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.84 int PRNDEC

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.85 int PRNDEF

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.86 int PRNTEX

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.87 int PRNWEB

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.88 double PRSOF

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.89 int PRVTX

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.90 double PSPLT[2]

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.91 double PTRMS

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.92 double PXRMS

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.93 double QC DL3

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.94 double QC DL5

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.95 double QC DLAM

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.96 double QDIQK

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.97 double QFCH[16]

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.98 double QG

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.99 double QSPAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.100 double QV

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.101 double SCABI

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.102 int SOFTME

Definition at line 70 of file HerwigWrapper6_4.h.

9.29.2.103 double SWEIN

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.104 double TLOUT

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.105 double TMTOP

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.106 double VCKM[3][3]

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.107 double VFCH[2][16]

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.108 double VGCUT

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.109 double VPCUT

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.110 double VQCUT

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.111 double WBIGST

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.112 double WGTMAX

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.113 double WGTSUM

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.114 double WSQSUM

Definition at line 40 of file HerwigWrapper6_4.h.

9.29.2.115 double ZBINM

Definition at line 62 of file HerwigWrapper6_4.h.

9.29.2.116 int ZPRIME

Definition at line 70 of file HerwigWrapper6_4.h.

9.30 initPythia.cc File Reference

```
#include "HepMC/PythiaWrapper.h"
```

Functions

- `void initPythia ()`

9.30.1 Function Documentation

9.30.1.1 `void initPythia ()`

Examples:

`example_MyPythia.cc`, `example_MyPythiaOnlyToHepMC.cc`, `example_PythiaStreamIO.cc`, and `testPythiaCopies.cc`.

Definition at line 11 of file `initPythia.cc`.

References `pydat2`, `pydatr`, `pypars`, and `pysubs`.

Referenced by `event_selection()`, `main()`, `pythia_in_out()`, `pythia_out()`, `pythia_particle_out()`, and `writePythiaStreamIO()`.

9.31 IO_AsciiParticles.cc File Reference

```
#include "HepMC/IO_AsciiParticles.h"  
#include "HepMC/GenEvent.h"  
#include "HepMC/ParticleDataTable.h"  
#include "HepMC/Version.h"
```

Namespaces

- namespace **HepMC**

9.32 IO_AsciiParticles.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_AsciiParticles**
event input/output in ascii format for eye and machine reading

9.33 IO_BaseClass.h File Reference

```
#include <iostream>
#include "HepMC/ParticleDataTable.h"
#include "HepMC/GenEvent.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_BaseClass**
all input/output classes inherit from IO_BaseClass (p. 158)

9.34 IO_Exception.h File Reference

```
#include <stdexcept>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_Exception**
IO exception handling.

9.35 IO_GenEvent.cc File Reference

```
#include "HepMC/IO_GenEvent.h"  
#include "HepMC/IO_Exception.h"  
#include "HepMC/GenEvent.h"  
#include "HepMC/ParticleDataTable.h"  
#include "HepMC/StreamHelpers.h"
```

Namespaces

- namespace **HepMC**

9.36 IO_GenEvent.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/IO_Exception.h"
#include "HepMC/Units.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_GenEvent**
IO_GenEvent (p. 164) also deals with *HeavyIon* (p. 133) and *PdfInfo* (p. 218).

9.37 IO_HEPEVT.cc File Reference

```
#include "HepMC/IO_HEPEVT.h"  
#include "HepMC/GenEvent.h"  
#include <cstdio>
```

Namespaces

- namespace **HepMC**

9.38 IO_HEPEVT.h File Reference

```
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_HEPEVT**
HEPEVT IO class.

9.39 IO_HERWIG.cc File Reference

```
#include "HepMC/IO_HERWIG.h"  
#include "HepMC/GenEvent.h"  
#include <cstdio>
```

Namespaces

- namespace **HepMC**

9.40 IO_HERWIG.h File Reference

```
#include <set>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_HERWIG**
IO_HERWIG (p. 174) is used to get Herwig information.

9.41 IO_PDG_ParticleDataTable.cc File Reference

```
#include <ctype.h>
#include <string>
#include <vector>
#include <cstdlib>
#include <cstring>
#include "HepMC/IO_PDG_ParticleDataTable.h"
```

Namespaces

- namespace **HepMC**

9.42 IO_PDG_ParticleDataTable.h File Reference

```
#include "HepMC/IO_BaseClass.h"  
#include <fstream>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_PDG_ParticleDataTable**
an example ParticleDataTable (p.211) IO method

9.43 is_arithmetic.h File Reference

Namespaces

- namespace **HepMC**
- namespace **detail**
- namespace **HepMC::detail**

Classes

- struct **HepMC::detail::is_arithmetic**< T >
undefined and therefore non-arithmetic
- struct **HepMC::detail::is_arithmetic**< char >
character is arithmetic
- struct **HepMC::detail::is_arithmetic**< unsigned char >
unsigned character is arithmetic
- struct **HepMC::detail::is_arithmetic**< signed char >
signed character is arithmetic
- struct **HepMC::detail::is_arithmetic**< short >
short is arithmetic
- struct **HepMC::detail::is_arithmetic**< unsigned short >
unsigned short is arithmetic
- struct **HepMC::detail::is_arithmetic**< int >
int is arithmetic
- struct **HepMC::detail::is_arithmetic**< unsigned int >
unsigned int is arithmetic
- struct **HepMC::detail::is_arithmetic**< long >
long is arithmetic
- struct **HepMC::detail::is_arithmetic**< unsigned long >
unsigned long is arithmetic
- struct **HepMC::detail::is_arithmetic**< float >
float is arithmetic
- struct **HepMC::detail::is_arithmetic**< double >
double is arithmetic
- struct **HepMC::detail::is_arithmetic**< long double >
long double is arithmetic

9.44 IsGoodEvent.h File Reference

Classes

- `class` **IsGoodEvent**
used in the tests

9.45 list_of_examples.cc File Reference

9.46 list_of_examples.cc File Reference

9.47 ParticleData.cc File Reference

```
#include "HepMC/ParticleData.h"  
#include <cstdio>
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &ostr, const ParticleData &pdata)`
write to ostr
- `double HepMC::clifetime_from_width (double width)`
set lifetime from width

9.48 ParticleData.h File Reference

```
#include <iostream>
#include <string>
#include <stdlib.h>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::ParticleData**
an example ParticleData (p. 204) class

Functions

- double **HepMC::lifetime_from_width** (double width)
set lifetime from width

Variables

- static const double **HepMC::HepMC_hbarc**
 *$\hbar c \rightarrow$ calculated with units of [mm*GeV]*

9.49 ParticleDataTable.h File Reference

```
#include <iostream>
#include <map>
#include <cstdio>
#include "HepMC/ParticleData.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::ParticleDataTable**
an example ParticleDataTable (p. 211) class

9.50 PdfInfo.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/PdfInfo.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &, PdfInfo const *)`
- `std::istream & HepMC::operator>> (std::istream &, PdfInfo *)`

9.51 PdfInfo.h File Reference

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::PdfInfo**
The PdfInfo (p. 218) class stores PDF information.

Functions

- **std::ostream & HepMC::operator<< (std::ostream &, PdfInfo const *)**
- **std::istream & HepMC::operator>> (std::istream &, PdfInfo *)**

9.52 Polarization.cc File Reference

```
#include "HepMC/Polarization.h"
```

Namespaces

- namespace **HepMC**

Functions

- `std::ostream & HepMC::operator<< (std::ostream &ostr, const Polarization &polar)`
print polarization information

9.53 Polarization.h File Reference

```
#include "HepMC/SimpleVector.h"  
#include <iostream>  
#include <cmath>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::Polarization**
The Polarization (p. 225) class stores theta and phi for a GenParticle (p. 99).

Variables

- static const double **HepMC::HepMC_pi = 3.14159265358979323846**

9.54 PythiaHelper.h File Reference

```
#include "HepMC/PythiaWrapper.h"
#include "HepMC/GenCrossSection.h"
```

Functions

- `void initPythia ()`
- `HepMC::GenCrossSection getPythiaCrossSection ()`
calculate the Pythia cross section and statistical error

9.54.1 Function Documentation

9.54.1.1 `HepMC::GenCrossSection getPythiaCrossSection ()` [inline]

calculate the Pythia cross section and statistical error

Examples:

`example_MyPythia.cc`, `example_MyPythiaOnlyToHepMC.cc`, `example_PythiaStreamIO.cc`,
and `testPythiaCopies.cc`.

Definition at line 16 of file `PythiaHelper.h`.

References `pyint5`, and `xsec`.

Referenced by `event_selection()`, `main()`, `pythia_in_out()`, `pythia_out()`, `pythia_particle_out()`, and `writePythiaStreamIO()`.

9.54.1.2 `void initPythia ()`

Definition at line 11 of file `initPythia.cc`.

References `pydat2`, `pydatr`, `pypars`, and `pysubs`.

Referenced by `event_selection()`, `main()`, `pythia_in_out()`, `pythia_out()`, `pythia_particle_out()`, and `writePythiaStreamIO()`.

9.55 PythiaWrapper.h File Reference

```
#include "HepMC/PythiaWrapper6_2.h"
```

9.56 PythiaWrapper5_720.h File Reference

```
#include <ctype.h>
```

Defines

- `#define initpydata initpydata_`
- `#define lujets lujets_`
- `#define ludat1 ludat1_`
- `#define ludat2 ludat2_`
- `#define ludat3 ludat3_`
- `#define ludatr ludatr_`
- `#define pysubs pysubs_`
- `#define pypars pypars_`
- `#define pyint1 pyint1_`
- `#define pyint2 pyint2_`
- `#define pyint5 pyint5_`
- `#define luhepc luhepc_`
- `#define pyinit pyinit_`
- `#define lulist lulist_`
- `#define pystat pystat_`
- `#define pyevnt pyevnt_`
- `#define ludata ludata_`
- `#define pydata pydata_`

Variables

- `struct {`
`int n`
`int k [5][pyjets_maxn]`
`float p [5][pyjets_maxn]`
`float v [5][pyjets_maxn]`
`} lujets_`
- `struct {`
`int mstu [200]`
`float paru [200]`
`int mstj [200]`
`float parj [200]`
`} ludat1_`
- `struct {`
`int kchg [3][500]`
`float pmas [4][500]`
`float parf [2000]`
`float vckm [4][4]`
`} ludat2_`

- struct {
 - int **mdecy** [3][500]
 - int **mdme** [2][2000]
 - float **brat** [2000]
 - int **kfdp** [5][2000]
 } **ludat3_**
- struct {
 - int **mrlu** [6]
 - float **rrlu** [100]
 } **ludatr_**
- struct {
 - int **msel**
 - int **msub** [200]
 - int **kfin** [81][2]
 - float **ckin** [200]
 } **pysubs_**
- struct {
 - int **mstp** [200]
 - float **parp** [200]
 - int **msti** [200]
 - float **pari** [200]
 } **pypars_**
- struct {
 - int **mint** [400]
 - float **vint** [400]
 } **pyint1_**
- struct {
 - int **iset** [200]
 - int **kfpr** [2][200]
 - float **coef** [20][200]
 - int **icol** [2][4][40]
 } **pyint2_**
- struct {
 - int **ngen** [3][201][3]
 - float **xsec** [3][201]
 } **pyint5_**

9.56.1 Define Documentation

9.56.1.1 #define initpydata initpydata_

Definition at line 26 of file PythiaWrapper5_720.h.

9.56.1.2 #define ludat1 ludat1_

Definition at line 47 of file PythiaWrapper5_720.h.

9.56.1.3 #define ludat2 ludat2_

Definition at line 53 of file PythiaWrapper5_720.h.

9.56.1.4 #define ludat3 ludat3_

Definition at line 60 of file PythiaWrapper5_720.h.

9.56.1.5 #define ludata ludata_

Definition at line 109 of file PythiaWrapper5_720.h.

9.56.1.6 #define ludatr ludatr_

Definition at line 66 of file PythiaWrapper5_720.h.

9.56.1.7 #define luhepc luhepc_

Definition at line 104 of file PythiaWrapper5_720.h.

9.56.1.8 #define lujets lujets_

Definition at line 39 of file PythiaWrapper5_720.h.

9.56.1.9 #define lulist lulist_

Definition at line 106 of file PythiaWrapper5_720.h.

9.56.1.10 #define pydata pydata_**9.56.1.11 #define pyevnt pyevnt_**

Definition at line 108 of file PythiaWrapper5_720.h.

9.56.1.12 #define pyinit pyinit_

Definition at line 105 of file PythiaWrapper5_720.h.

9.56.1.13 #define pyint1 pyint1_

Definition at line 86 of file PythiaWrapper5_720.h.

9.56.1.14 #define pyint2 pyint2_

Definition at line 93 of file PythiaWrapper5_720.h.

9.56.1.15 #define pyint5 pyint5_

Definition at line 99 of file PythiaWrapper5_720.h.

Referenced by `getPythiaCrossSection()`.

9.56.1.16 #define pypars pypars_**Examples:**

**example_MyPythia.cc, example_MyPythiaOnlyToHepMC.cc, example_PythiaStreamIO.cc,
and testPythiaCopies.cc.**

Definition at line 80 of file PythiaWrapper5_720.h.

Referenced by `event_selection()`, `initPythia()`, `main()`, `pythia_out()`,
and `writePythiaStreamIO()`.

9.56.1.17 #define pystat pystat_

Definition at line 107 of file PythiaWrapper5_720.h.

9.56.1.18 #define pysubs pysubs_

Definition at line 72 of file PythiaWrapper5_720.h.

Referenced by `initPythia()`.

9.56.2 Variable Documentation**9.56.2.1 float brat[2000]**

Definition at line 57 of file PythiaWrapper5_720.h.

9.56.2.2 float ckin[200]

Definition at line 70 of file PythiaWrapper5_720.h.

9.56.2.3 float coef[20][200]

Definition at line 90 of file PythiaWrapper5_720.h.

9.56.2.4 int icol[2][4][40]

Definition at line 91 of file PythiaWrapper5_720.h.

9.56.2.5 int iset[200]

Definition at line 89 of file PythiaWrapper5_720.h.

9.56.2.6 int k[5][pyjets_maxn]

Definition at line 36 of file PythiaWrapper5_720.h.

9.56.2.7 int kchg[3][500]

Definition at line 50 of file PythiaWrapper5_720.h.

9.56.2.8 int kfdp[5][2000]

Definition at line 58 of file PythiaWrapper5_720.h.

9.56.2.9 int kfin[81][2]

Definition at line 69 of file PythiaWrapper5_720.h.

9.56.2.10 int kfpr[2][200]

Definition at line 89 of file PythiaWrapper5_720.h.

9.56.2.11 struct { ... } ludat1_**9.56.2.12 struct { ... } ludat2_****9.56.2.13 struct { ... } ludat3_****9.56.2.14 struct { ... } ludatr_****9.56.2.15 struct { ... } lujets_****9.56.2.16 int mdcy[3][500]**

Definition at line 56 of file PythiaWrapper5_720.h.

9.56.2.17 int mdme[2][2000]

Definition at line 56 of file PythiaWrapper5_720.h.

9.56.2.18 int mint[400]

Definition at line 83 of file PythiaWrapper5_720.h.

9.56.2.19 int mrlu[6]

Definition at line 63 of file PythiaWrapper5_720.h.

9.56.2.20 int msel

Definition at line 69 of file PythiaWrapper5_720.h.

9.56.2.21 int msti[200]

Definition at line 77 of file PythiaWrapper5_720.h.

9.56.2.22 int mstj[200]

Definition at line 44 of file PythiaWrapper5_720.h.

9.56.2.23 int mstp[200]

Definition at line 75 of file PythiaWrapper5_720.h.

9.56.2.24 int mstu[200]

Definition at line 42 of file PythiaWrapper5_720.h.

9.56.2.25 int msub[200]

Definition at line 69 of file PythiaWrapper5_720.h.

9.56.2.26 int n

Definition at line 36 of file PythiaWrapper5_720.h.

9.56.2.27 int ngen[3][201][3]

Definition at line 96 of file PythiaWrapper5_720.h.

9.56.2.28 float p[5][pyjets_maxn]**Examples:**

example_BuildEventFromScratch.cc, example_EventSelection.cc, example_MyPythia.cc, and example_UsingIterators.cc.

Definition at line 37 of file PythiaWrapper5_720.h.

Referenced by `HepMC::TempParticleMap::addEndParticle()`, `HepMC::already_in_vector()`, `HepMC::IO_HERWIG::build_end_vertex()`, `HepMC::IO_HEPEVT::build_end_vertex()`, `HepMC::IO_HERWIG::build_particle()`,

```
HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_
production_vertex(), HepMC::IO_HEPEVT::build_production_vertex(),
HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_
partners(), HepMC::GenVertex::edge_iterator::edge_iterator(),
HepMC::TempParticleMap::end_vertex(), HepMC::IO_HERWIG::find_
in_map(), HepMC::IO_HEPEVT::find_in_map(), findPiZero(), Hep-
MC::GenEvent::GenEvent(), IsPhoton(), IsWBoson(), main(), Hep-
MC::ParticleDataTable::make_antiparticles_from_particles(), Hep-
MC::ParticleDataTable::merge_table(), HepMC::not_in_vector(), Is-
FinalState::operator()(), IsGoodEvent::operator()(), IsState-
Final::operator()(), IsW_Boson::operator()(), IsPhoton::operator()(),
IsGoodEventMyPythia::operator()(), IsEventGood::operator()(), Hep-
MC::GenVertex::edge_iterator::operator=(), particleTypes(), Hep-
MC::GenEvent::read(), HepMC::detail::read_particle(), HepMC::Gen-
Event::remove_barcode(), HepMC::GenEvent::set_barcode(), HepMC::Gen-
Event::set_pdf_info(), HepMC::GenEvent::valid_beam_particles(), and
HepMC::IO_HEPEVT::write_event().
```

9.56.2.29 float parf[2000]

Definition at line 51 of file PythiaWrapper5_720.h.

9.56.2.30 float pari[200]

Definition at line 78 of file PythiaWrapper5_720.h.

9.56.2.31 float parj[200]

Definition at line 45 of file PythiaWrapper5_720.h.

9.56.2.32 float parp[200]

Definition at line 76 of file PythiaWrapper5_720.h.

9.56.2.33 float paru[200]

Definition at line 43 of file PythiaWrapper5_720.h.

9.56.2.34 float pmas[4][500]

Definition at line 51 of file PythiaWrapper5_720.h.

9.56.2.35 struct { ... } pyint1_

9.56.2.36 struct { ... } pyint2_

9.56.2.37 struct { ... } pyint5_

9.56.2.38 struct { ... } pypars_

9.56.2.39 struct { ... } pysubs_

9.56.2.40 float rrlu[100]

Definition at line 64 of file PythiaWrapper5_720.h.

9.56.2.41 float v[5][pyjets_maxn]

Examples:

example_UsingIterators.cc, testHepMCIteration.cc.in, and VectorConversion.h.

Definition at line 37 of file PythiaWrapper5_720.h.

Referenced by HepMC::compareVertices(), convertTo(), HepMC::GenEvent::GenEvent(), main(), HepMC::GenEvent::read(), HepMC::detail::read_vertex(), HepMC::GenEvent::remove_barcode(), HepMC::GenEvent::set_barcode(), HepMC::GenEvent::write(), and HepMC::IO_HEPEVT::write_event().

9.56.2.42 float vckm[4][4]

Definition at line 51 of file PythiaWrapper5_720.h.

9.56.2.43 float vint[400]

Definition at line 84 of file PythiaWrapper5_720.h.

9.56.2.44 float xsec[3][201]

Definition at line 97 of file PythiaWrapper5_720.h.

Referenced by getHerwigCrossSection(), and getPythiaCrossSection().

9.57 PythiaWrapper6_152.h File Reference

```
#include <ctype.h>
#include <cstring>
```

Defines

- `#define initpydata initpydata_`
- `#define pyjets pyjets_`
- `#define pydat1 pydat1_`
- `#define pydat2 pydat2_`
- `#define pydat3 pydat3_`
- `#define pydatr pydatr_`
- `#define pysubs pysubs_`
- `#define pypars pypars_`
- `#define pyint1 pyint1_`
- `#define pyint2 pyint2_`
- `#define pyint5 pyint5_`
- `#define pyhepc pyhepc_`
- `#define pyinit pyinit_`
- `#define pylist pylist_`
- `#define pystat pystat_`
- `#define pyevnt pyevnt_`
- `#define pydata pydata_`

Variables

- `struct {`
 `int n`
 `int npad`
 `int k [5][pyjets_maxn]`
 `double p [5][pyjets_maxn]`
 `double v [5][pyjets_maxn]`
 `} pyjets_`
- `struct {`
 `int mstu [200]`
 `double paru [200]`
 `int mstj [200]`
 `double parj [200]`
 `} pydat1_`
- `struct {`
 `int kchg [4][500]`
 `double pmas [4][500]`
 `double parf [2000]`
 `double vckm [4][4]`
 `} pydat2_`

- struct {
 int **mdecy** [3][500]
 int **mdme** [2][4000]
 double **brat** [4000]
 int **kfdp** [5][4000]
 } **pydat3_**
- struct {
 int **mrpy** [6]
 double **rrpy** [100]
 } **pydatr_**
- struct {
 int **msel**
 int **mselpd**
 int **msub** [500]
 int **kfin** [81][2]
 double **ckin** [200]
 } **pysubs_**
- struct {
 int **mstp** [200]
 double **parp** [200]
 int **msti** [200]
 double **pari** [200]
 } **pypars_**
- struct {
 int **mint** [400]
 double **vint** [400]
 } **pyint1_**
- struct {
 int **iset** [500]
 int **kfpr** [2][500]
 double **coef** [20][500]
 int **icol** [2][4][40]
 } **pyint2_**
- struct {
 int **ngenpd**
 int **ngen** [3][501]
 double **xsec** [3][501]
 } **pyint5_**

9.57.1 Define Documentation

9.57.1.1 #define initpydata initpydata_

Definition at line 27 of file PythiaWrapper6_152.h.

9.57.1.2 #define pydat1 pydat1_

Definition at line 48 of file PythiaWrapper6_152.h.

9.57.1.3 #define pydat2 pydat2_

Definition at line 54 of file PythiaWrapper6_152.h.

Referenced by `initPythia()`.

9.57.1.4 #define pydat3 pydat3_

Definition at line 61 of file PythiaWrapper6_152.h.

9.57.1.5 #define pydata pydata_**9.57.1.6 #define pydatr pydatr_**

Definition at line 67 of file PythiaWrapper6_152.h.

Referenced by `initPythia()`.

9.57.1.7 #define pyevnt pyevnt_

Definition at line 109 of file PythiaWrapper6_152.h.

9.57.1.8 #define pyhepc pyhepc_

Definition at line 105 of file PythiaWrapper6_152.h.

9.57.1.9 #define pyinit pyinit_

Definition at line 106 of file PythiaWrapper6_152.h.

9.57.1.10 #define pyint1 pyint1_

Definition at line 87 of file PythiaWrapper6_152.h.

9.57.1.11 #define pyint2 pyint2_

Definition at line 94 of file PythiaWrapper6_152.h.

9.57.1.12 #define pyint5 pyint5_

Definition at line 100 of file PythiaWrapper6_152.h.

9.57.1.13 #define pyjets pyjets_

Definition at line 40 of file PythiaWrapper6_152.h.

9.57.1.14 #define pylist pylist_

Definition at line 107 of file PythiaWrapper6_152.h.

9.57.1.15 #define pypars pypars_

Definition at line 81 of file PythiaWrapper6_152.h.

9.57.1.16 #define pystat pystat_

Definition at line 108 of file PythiaWrapper6_152.h.

9.57.1.17 #define pysubs pysubs_

Definition at line 73 of file PythiaWrapper6_152.h.

9.57.2 Variable Documentation**9.57.2.1 double brat[4000]**

Definition at line 58 of file PythiaWrapper6_152.h.

9.57.2.2 double ckin[200]

Definition at line 71 of file PythiaWrapper6_152.h.

9.57.2.3 double coef[20][500]

Definition at line 91 of file PythiaWrapper6_152.h.

9.57.2.4 int icol[2][4][40]

Definition at line 92 of file PythiaWrapper6_152.h.

9.57.2.5 int iset[500]

Definition at line 90 of file PythiaWrapper6_152.h.

9.57.2.6 int k[5][pyjets_maxn]

Definition at line 37 of file PythiaWrapper6_152.h.

9.57.2.7 int kchg[4][500]

Definition at line 51 of file PythiaWrapper6_152.h.

9.57.2.8 int kfdp[5][4000]

Definition at line 59 of file PythiaWrapper6_152.h.

9.57.2.9 int kfin[81][2]

Definition at line 70 of file PythiaWrapper6_152.h.

9.57.2.10 int kfpr[2][500]

Definition at line 90 of file PythiaWrapper6_152.h.

9.57.2.11 int mdcy[3][500]

Definition at line 57 of file PythiaWrapper6_152.h.

9.57.2.12 int mdme[2][4000]

Definition at line 57 of file PythiaWrapper6_152.h.

9.57.2.13 int mint[400]

Definition at line 84 of file PythiaWrapper6_152.h.

9.57.2.14 int mrpy[6]

Definition at line 64 of file PythiaWrapper6_152.h.

9.57.2.15 int msel

Definition at line 70 of file PythiaWrapper6_152.h.

9.57.2.16 int mselpd

Definition at line 70 of file PythiaWrapper6_152.h.

9.57.2.17 int msti[200]

Definition at line 78 of file PythiaWrapper6_152.h.

9.57.2.18 int mstj[200]

Definition at line 45 of file PythiaWrapper6_152.h.

9.57.2.19 int mstp[200]

Definition at line 76 of file PythiaWrapper6_152.h.

9.57.2.20 int mstu[200]

Definition at line 43 of file PythiaWrapper6_152.h.

9.57.2.21 int msub[500]

Definition at line 70 of file PythiaWrapper6_152.h.

9.57.2.22 int n

Definition at line 37 of file PythiaWrapper6_152.h.

9.57.2.23 int ngen[3][501]

Definition at line 97 of file PythiaWrapper6_152.h.

9.57.2.24 int ngenpd

Definition at line 97 of file PythiaWrapper6_152.h.

9.57.2.25 int npad

Definition at line 37 of file PythiaWrapper6_152.h.

9.57.2.26 double p[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_152.h.

9.57.2.27 double parf[2000]

Definition at line 52 of file PythiaWrapper6_152.h.

9.57.2.28 double pari[200]

Definition at line 79 of file PythiaWrapper6_152.h.

9.57.2.29 double parj[200]

Definition at line 46 of file PythiaWrapper6_152.h.

9.57.2.30 double parp[200]

Definition at line 77 of file PythiaWrapper6_152.h.

9.57.2.31 double paru[200]

Definition at line 44 of file PythiaWrapper6_152.h.

9.57.2.32 double pmas[4][500]

Definition at line 52 of file PythiaWrapper6_152.h.

9.57.2.33 struct { ... } pydat1_**9.57.2.34 struct { ... } pydat2_****9.57.2.35 struct { ... } pydat3_****9.57.2.36 struct { ... } pydatr_****9.57.2.37 struct { ... } pyint1_****9.57.2.38 struct { ... } pyint2_****9.57.2.39 struct { ... } pyint5_****9.57.2.40 struct { ... } pyjets_****9.57.2.41 struct { ... } pypars_****9.57.2.42 struct { ... } pysubs_****9.57.2.43 double rrp[100]**

Definition at line 65 of file PythiaWrapper6_152.h.

9.57.2.44 double v[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_152.h.

9.57.2.45 double vckm[4][4]

Definition at line 52 of file PythiaWrapper6_152.h.

9.57.2.46 double vint[400]

Definition at line 85 of file PythiaWrapper6_152.h.

9.57.2.47 double xsec[3][501]

Definition at line 98 of file PythiaWrapper6_152.h.

9.58 PythiaWrapper6_152_WIN32.h File Reference

9.59 PythiaWrapper6_2.h File Reference

```
#include <ctype.h>
#include <cstring>
```

Defines

- `#define initpydata initpydata_`
- `#define pyjets pyjets_`
- `#define pydat1 pydat1_`
- `#define pydat2 pydat2_`
- `#define pydat3 pydat3_`
- `#define pydatr pydatr_`
- `#define pysubs pysubs_`
- `#define pypars pypars_`
- `#define pyint1 pyint1_`
- `#define pyint2 pyint2_`
- `#define pyint5 pyint5_`
- `#define pyhepc pyhepc_`
- `#define pyinit pyinit_`
- `#define pylist pylist_`
- `#define pystate pystate_`
- `#define pyevnt pyevnt_`
- `#define upinit upinit_`
- `#define upevnt upevnt_`
- `#define pydata pydata_`

Functions

- `void initpydata (void)`

Variables

- `const int pyjets_maxn = 4000`
- `struct {`
 - `int n`
 - `int npad`
 - `int k [5][pyjets_maxn]`
 - `double p [5][pyjets_maxn]`
 - `double v [5][pyjets_maxn]``} pyjets_`
- `struct {`
 - `int mstu [200]`
 - `double paru [200]`
 - `int mstj [200]`
 - `double parj [200]``} pydat1_`

- struct {
 int kchg [4][500]
 double pmas [4][500]
 double parf [2000]
 double vckm [4][4]
} pydat2_
- struct {
 int mdcy [3][500]
 int mdme [2][8000]
 double brat [8000]
 int kfdp [5][8000]
} pydat3_
- struct {
 int mrpy [6]
 double rrp [100]
} pydatr_
- struct {
 int msel
 int mselpd
 int msub [500]
 int kfin [81][2]
 double ckin [200]
} pysubs_
- struct {
 int mstp [200]
 double parp [200]
 int msti [200]
 double pari [200]
} pypars_
- struct {
 int mint [400]
 double vint [400]
} pyint1_
- struct {
 int iset [500]
 int kfpr [2][500]
 double coef [20][500]
 int icol [2][4][40]
} pyint2_
- struct {
 int ngenpd
 int ngen [3][501]
 double xsec [3][501]
} pyint5_

9.59.1 Define Documentation

9.59.1.1 #define initpydata initpydata_

Definition at line 30 of file PythiaWrapper6_2.h.

9.59.1.2 #define pydat1 pydat1_

Definition at line 52 of file PythiaWrapper6_2.h.

9.59.1.3 #define pydat2 pydat2_

Definition at line 60 of file PythiaWrapper6_2.h.

9.59.1.4 #define pydat3 pydat3_

Definition at line 69 of file PythiaWrapper6_2.h.

9.59.1.5 #define pydata pydata_

9.59.1.6 #define pydatr pydatr_

Definition at line 77 of file PythiaWrapper6_2.h.

9.59.1.7 #define pyevnt pyevnt_

Definition at line 129 of file PythiaWrapper6_2.h.

9.59.1.8 #define pyhepc pyhepc_

Definition at line 125 of file PythiaWrapper6_2.h.

9.59.1.9 #define pyinit pyinit_

Definition at line 126 of file PythiaWrapper6_2.h.

9.59.1.10 #define pyint1 pyint1_

Definition at line 103 of file PythiaWrapper6_2.h.

9.59.1.11 #define pyint2 pyint2_

Definition at line 112 of file PythiaWrapper6_2.h.

9.59.1.12 #define pyint5 pyint5_

Definition at line 120 of file PythiaWrapper6_2.h.

9.59.1.13 #define pyjets pyjets_

Definition at line 42 of file PythiaWrapper6_2.h.

9.59.1.14 #define pylist pylist_

Definition at line 127 of file PythiaWrapper6_2.h.

9.59.1.15 #define pypars pypars_

Definition at line 95 of file PythiaWrapper6_2.h.

9.59.1.16 #define pystat pystat_

Definition at line 128 of file PythiaWrapper6_2.h.

9.59.1.17 #define pysubs pysubs_

Definition at line 85 of file PythiaWrapper6_2.h.

9.59.1.18 #define upevnt upevnt_

Definition at line 131 of file PythiaWrapper6_2.h.

9.59.1.19 #define upinit upinit_

Definition at line 130 of file PythiaWrapper6_2.h.

9.59.2 Function Documentation**9.59.2.1 void initpydata (void)****9.59.3 Variable Documentation****9.59.3.1 double brat[8000]**

Definition at line 65 of file PythiaWrapper6_2.h.

9.59.3.2 double ckin[200]

Definition at line 82 of file PythiaWrapper6_2.h.

9.59.3.3 double coef[20][500]

Definition at line 108 of file PythiaWrapper6_2.h.

9.59.3.4 int icol[2][4][40]

Definition at line 109 of file PythiaWrapper6_2.h.

9.59.3.5 int iset[500]

Definition at line 107 of file PythiaWrapper6_2.h.

9.59.3.6 int k[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_2.h.

9.59.3.7 int kchg[4][500]

Definition at line 56 of file PythiaWrapper6_2.h.

9.59.3.8 int kfdp[5][8000]

Definition at line 66 of file PythiaWrapper6_2.h.

9.59.3.9 int kfin[81][2]

Definition at line 81 of file PythiaWrapper6_2.h.

9.59.3.10 int kfpr[2][500]

Definition at line 107 of file PythiaWrapper6_2.h.

9.59.3.11 int mdcy[3][500]

Definition at line 64 of file PythiaWrapper6_2.h.

9.59.3.12 int mdme[2][8000]

Definition at line 64 of file PythiaWrapper6_2.h.

9.59.3.13 int mint[400]

Definition at line 99 of file PythiaWrapper6_2.h.

9.59.3.14 int mrpy[6]

Definition at line 73 of file PythiaWrapper6_2.h.

9.59.3.15 int msel

Definition at line 81 of file PythiaWrapper6_2.h.

9.59.3.16 int mselpd

Definition at line 81 of file PythiaWrapper6_2.h.

9.59.3.17 int msti[200]

Definition at line 91 of file PythiaWrapper6_2.h.

9.59.3.18 int mstj[200]

Definition at line 48 of file PythiaWrapper6_2.h.

9.59.3.19 int mstp[200]

Definition at line 89 of file PythiaWrapper6_2.h.

9.59.3.20 int mstu[200]

Definition at line 46 of file PythiaWrapper6_2.h.

9.59.3.21 int msub[500]

Definition at line 81 of file PythiaWrapper6_2.h.

9.59.3.22 int n

Definition at line 38 of file PythiaWrapper6_2.h.

9.59.3.23 int ngen[3][501]

Definition at line 116 of file PythiaWrapper6_2.h.

9.59.3.24 int ngenpd

Definition at line 116 of file PythiaWrapper6_2.h.

9.59.3.25 int npad

Definition at line 38 of file PythiaWrapper6_2.h.

9.59.3.26 double p[5][pyjets_maxn]

Definition at line 39 of file PythiaWrapper6_2.h.

9.59.3.27 double parf[2000]

Definition at line 57 of file PythiaWrapper6_2.h.

9.59.3.28 double pari[200]

Definition at line 92 of file PythiaWrapper6_2.h.

9.59.3.29 double parj[200]

Definition at line 49 of file PythiaWrapper6_2.h.

9.59.3.30 double parp[200]

Definition at line 90 of file PythiaWrapper6_2.h.

9.59.3.31 double paru[200]

Definition at line 47 of file PythiaWrapper6_2.h.

9.59.3.32 double pmas[4][500]

Definition at line 57 of file PythiaWrapper6_2.h.

9.59.3.33 struct { ... } pydat1_**9.59.3.34 struct { ... } pydat2_****9.59.3.35 struct { ... } pydat3_****9.59.3.36 struct { ... } pydatr_****9.59.3.37 struct { ... } pyint1_****9.59.3.38 struct { ... } pyint2_****9.59.3.39 struct { ... } pyint5_****9.59.3.40 struct { ... } pyjets_****9.59.3.41 const int pyjets_maxn = 4000**

Definition at line 35 of file PythiaWrapper6_2.h.

9.59.3.42 struct { ... } pypars_**9.59.3.43 struct { ... } pysubs_****9.59.3.44 double rrp[100]**

Definition at line 74 of file PythiaWrapper6_2.h.

9.59.3.45 double v[5][pyjets_maxn]

Definition at line 39 of file PythiaWrapper6_2.h.

9.59.3.46 double vckm[4][4]

Definition at line 57 of file PythiaWrapper6_2.h.

9.59.3.47 double vint[400]

Definition at line 100 of file PythiaWrapper6_2.h.

9.59.3.48 double xsec[3][501]

Definition at line 117 of file PythiaWrapper6_2.h.

9.60 PythiaWrapper6_2_WIN32.h File Reference

9.61 SearchVector.cc File Reference

```
#include "HepMC/SearchVector.h"
```

Namespaces

- namespace **HepMC**

Functions

- `bool HepMC::not_in_vector (std::vector< HepMC::GenParticle * > *, GenParticle *)`
returns true if it cannot find GenParticle in the vector*
- `std::vector< HepMC::GenParticle * >::iterator HepMC::already_in_vector (std::vector< GenParticle * > *v, GenParticle *p)`
returns true if GenParticle (p.99) is in the vector

9.62 SearchVector.h File Reference

```
#include "HepMC/GenVertex.h"  
#include "HepMC/GenParticle.h"
```

Namespaces

- namespace **HepMC**

Functions

- **bool HepMC::not_in_vector** (std::vector< HepMC::GenParticle * > *, GenParticle *)
returns true if it cannot find GenParticle in the vector*
- **std::vector< HepMC::GenParticle * >::iterator HepMC::already_in_vector** (std::vector< GenParticle * > *v, GenParticle *p)
returns true if GenParticle (p.99) is in the vector

9.63 SimpleVector.h File Reference

```
#include "HepMC/enable_if.h"
#include "HepMC/is_arithmetic.h"
#include "HepMC/SimpleVector.icc"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::FourVector**
FourVector (p. 50) is a simple representation of a physics 4 vector.
- class **HepMC::ThreeVector**
ThreeVector (p. 238) is a simple representation of a position or displacement 3 vector.

9.64 StreamHelpers.cc File Reference

```
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- namespace **HepMC**
- namespace **HepMC::detail**

Functions

- `std::istream & HepMC::detail::read_vertex (std::istream &, TempParticleMap &, GenVertex *)`
- `std::istream & HepMC::detail::find_event_end (std::istream &)`
used to read to the end of a bad event

9.65 StreamHelpers.h File Reference

```
#include <ostream>
#include <istream>
#include "HepMC/GenEvent.h"
#include "HepMC/TempParticleMap.h"
```

Namespaces

- namespace **HepMC**
- namespace **HepMC::detail**

Functions

- **std::ostream & HepMC::detail::establish_output_stream_info (std::ostream &)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & HepMC::detail::establish_input_stream_info (std::istream &)**
used by IO_GenEvent (p. 164) constructor
- **std::istream & HepMC::detail::read_vertex (std::istream &, TempParticleMap &, GenVertex *)**
- **std::istream & HepMC::detail::read_particle (std::istream &, TempParticleMap &, GenParticle *)**
- **std::ostream & HepMC::detail::output (std::ostream &os, const double &d)**
write a double - for internal use by streaming IO
- **std::ostream & HepMC::detail::output (std::ostream &os, const float &d)**
write a float - for internal use by streaming IO
- **std::ostream & HepMC::detail::output (std::ostream &os, const int &i)**
write an int - for internal use by streaming IO
- **std::ostream & HepMC::detail::output (std::ostream &os, const long &i)**
write a long - for internal use by streaming IO
- **std::ostream & HepMC::detail::output (std::ostream &os, const char &c)**
write a single char - for internal use by streaming IO
- **std::istream & HepMC::detail::find_event_end (std::istream &)**
used to read to the end of a bad event

9.66 StreamInfo.cc File Reference

```
#include <string>
#include "HepMC/StreamInfo.h"
```

Namespaces

- namespace **HepMC**

9.67 StreamInfo.h File Reference

```
#include <string>
#include "HepMC/Units.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::StreamInfo**
StreamInfo (p. 230) contains extra information needed when using streaming IO.

Enumerations

- enum **HepMC::known_io** {
 HepMC::gen = 1, HepMC::ascii, HepMC::extascii, HepMC::ascii_pdt,
 HepMC::extascii_pdt }
 The known_io enum is used to track which type of input is being read.

9.68 TempParticleMap.h File Reference

```
#include <map>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::TempParticleMap**
TempParticleMap (p. 235) is a temporary *GenParticle** container used during input.

9.69 testFlow.cc File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Typedefs

- typedef std::vector< **HepMC::GenParticle *** > **FlowVec**

Functions

- int **main** ()

9.69.1 Typedef Documentation

9.69.1.1 typedef std::vector<HepMC::GenParticle*> FlowVec

Definition at line 15 of file testFlow.cc.

9.69.2 Function Documentation

9.69.2.1 int main ()

Definition at line 17 of file testFlow.cc.

References `HepMC::GenVertex::add_particle_in()`, `HepMC::GenVertex::add_particle_out()`, `HepMC::GenEvent::add_vertex()`, `HepMC::GenParticle::barcode()`, `HepMC::Flow::erase()`, `HepMC::GenParticle::flow()`, `HepMC::GenEvent::print()`, `HepMC::GenParticle::set_flow()`, and `HepMC::GenEvent::set_signal_process_vertex()`.

9.70 testHepMCIteration.h File Reference

Classes

- `class IsFinalState`
test class

Functions

- `bool IsPhoton (const HepMC::GenParticle *p)`
returns true if the GenParticle particle is a photon with $p_T > 10$ GeV
- `bool IsWBoson (const HepMC::GenParticle *p)`
returns true if the GenParticle is a W^+/W^-

9.70.1 Function Documentation

9.70.1.1 `bool IsPhoton (const HepMC::GenParticle *p)`

returns true if the GenParticle particle is a photon with $p_T > 10$ GeV

Examples:

`testHepMCIteration.cc.in.`

Definition at line 10 of file testHepMCIteration.h.

References `p`.

9.70.1.2 `bool IsWBoson (const HepMC::GenParticle *p)`

returns true if the GenParticle is a W^+/W^-

Examples:

`testHepMCIteration.cc.in.`

Definition at line 17 of file testHepMCIteration.h.

References `p`.

9.71 testHepMCMethods.cc File Reference

```
#include "testHepMCMethods.h"
```

Functions

- `double findPiZero (HepMC::GenEvent *evt)`
- `void particleTypes (HepMC::GenEvent *evt)`

9.71.1 Function Documentation

9.71.1.1 `double findPiZero (HepMC::GenEvent * evt)`

Examples:

`testHepMC.cc.in`, and `testStreamIO.cc.in`.

Definition at line 11 of file `testHepMCMethods.cc`.

References `p`, `HepMC::GenEvent::particles_begin()`, and `HepMC::GenEvent::particles_end()`.

9.71.1.2 `void particleTypes (HepMC::GenEvent * evt)`

Examples:

`testHepMC.cc.in`, and `testStreamIO.cc.in`.

Definition at line 22 of file `testHepMCMethods.cc`.

References `HepMC::GenEvent::event_number()`, `p`, `HepMC::GenEvent::particles_begin()`, `HepMC::GenEvent::particles_end()`, and `HepMC::GenEvent::particles_size()`.

9.72 testHepMCMethods.h File Reference

```
#include "HepMC/GenEvent.h"
```

Functions

- `double findPiZero (HepMC::GenEvent *)`
- `void particleTypes (HepMC::GenEvent *)`

9.72.1 Function Documentation

9.72.1.1 `double findPiZero (HepMC::GenEvent *)`

Definition at line 11 of file testHepMCMethods.cc.

References `p`, `HepMC::GenEvent::particles_begin()`, and `HepMC::GenEvent::particles_end()`.

9.72.1.2 `void particleTypes (HepMC::GenEvent *)`

Definition at line 22 of file testHepMCMethods.cc.

References `HepMC::GenEvent::event_number()`, `p`, `HepMC::GenEvent::particles_begin()`, `HepMC::GenEvent::particles_end()`, and `HepMC::GenEvent::particles_size()`.

9.73 testHerwigCopies.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
#include "HerwigHelper.h"
```

Functions

- `int main()`

9.73.1 Function Documentation

9.73.1.1 `int main()`

Definition at line 17 of file testHerwigCopies.cc.

References `HepMC::compareGenEvent()`, `HepMC::GenEvent::event_number()`, `getHerwigCrossSection()`, `HepMC::Units::GEV`, `hwbgen`, `hwbmch`, `hwcdec`, `hwcfor`, `hwdhad`, `hwdhob`, `hwdhvy`, `hwefin`, `hweini`, `hwepro`, `hwevnt`, `hwigin`, `hwmevt`, `hwproc`, `hwufne`, `hwuinc`, `hwuine`, `HepMC::Units::MM`, `HepMC::GenEvent::print()`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::GenEvent::set_event_number()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_signal_process_id()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, and `HepMC::GenEvent::use_units()`.

9.74 testPrintBug.cc File Reference

```
#include <fstream>
#include "HepMC/GenEvent.h"
#include "HepMC/SimpleVector.h"
```

Functions

- `int main (int argc, char *argv[])`

9.74.1 Function Documentation

9.74.1.1 `int main (int argc, char * argv[])`

Definition at line 10 of file testPrintBug.cc.

References `HepMC::GenVertex::add_particle_in()`, `HepMC::GenVertex::add_particle_out()`, `HepMC::GenEvent::add_vertex()`, and `HepMC::GenEvent::print()`.

9.75 testPythiaCopies.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "PythiaHelper.h"
```

Functions

- `int main()`

9.75.1 Function Documentation

9.75.1.1 `int main()`

Definition at line 16 of file testPythiaCopies.cc.

References `HepMC::compareGenEvent()`, `HepMC::GenEvent::event_number()`, `getPythiaCrossSection()`, `HepMC::Units::GEV`, `initPythia()`, `HepMC::Units::MM`, `HepMC::GenEvent::print()`, `pypars`, `HepMC::IO_BaseClass::read_next_event()`, `HepMC::GenEvent::set_cross_section()`, `HepMC::HEPEVT_Wrapper::set_max_number_entries()`, `HepMC::GenEvent::set_mpi()`, `HepMC::HEPEVT_Wrapper::set_sizeof_real()`, and `HepMC::GenEvent::use_units()`.

9.76 testSimpleVector.cc File Reference

```
#include <iostream>
#include "HepMC/SimpleVector.h"
```

Functions

- `int main()`

9.76.1 Function Documentation

9.76.1.1 `int main()`

Definition at line 8 of file testSimpleVector.cc.

References `HepMC::FourVector::e()`, `HepMC::FourVector::eta()`, `HepMC::FourVector::m()`, `HepMC::FourVector::m2()`, `HepMC::FourVector::perp()`, `HepMC::ThreeVector::perp()`, `HepMC::FourVector::perp2()`, `HepMC::ThreeVector::perp2()`, `HepMC::FourVector::phi()`, `HepMC::ThreeVector::phi()`, `HepMC::FourVector::pseudoRapidity()`, `HepMC::FourVector::px()`, `HepMC::FourVector::py()`, `HepMC::FourVector::pz()`, `HepMC::ThreeVector::r()`, `HepMC::FourVector::rho()`, `HepMC::FourVector::set()`, `HepMC::ThreeVector::set()`, `HepMC::FourVector::setE()`, `HepMC::ThreeVector::setPhi()`, `HepMC::FourVector::setPx()`, `HepMC::FourVector::setPy()`, `HepMC::FourVector::setPz()`, `HepMC::FourVector::setT()`, `HepMC::ThreeVector::setTheta()`, `HepMC::FourVector::setX()`, `HepMC::ThreeVector::setX()`, `HepMC::FourVector::setY()`, `HepMC::ThreeVector::setY()`, `HepMC::FourVector::setZ()`, `HepMC::ThreeVector::setZ()`, `HepMC::FourVector::t()`, `HepMC::FourVector::theta()`, `HepMC::ThreeVector::theta()`, `HepMC::FourVector::x()`, `HepMC::ThreeVector::x()`, `HepMC::FourVector::y()`, `HepMC::ThreeVector::y()`, `HepMC::FourVector::z()`, and `HepMC::ThreeVector::z()`.

9.77 testUnits.cc File Reference

```
#include <iostream>
#include "HepMC/Units.h"
```

Functions

- `int main()`

9.77.1 Function Documentation

9.77.1.1 `int main()`

Definition at line 8 of file testUnits.cc.

References `HepMC::Units::CM`, `HepMC::Units::conversion_factor()`, `HepMC::Units::default_length_unit()`, `HepMC::Units::default_momentum_unit()`, `HepMC::Units::GEV`, `HepMC::Units::MEV`, `HepMC::Units::MM`, and `HepMC::Units::name()`.

9.78 Units.h File Reference

```
#include <iostream>
#include <string>
```

Namespaces

- namespace **HepMC**
- namespace **Units**
- namespace **HepMC::Units**

Enumerations

- enum **HepMC::Units::MomentumUnit** { **HepMC::Units::MEV**, **HepMC::Units::GEV** }
- enum **HepMC::Units::LengthUnit** { **HepMC::Units::MM**, **HepMC::Units::CM** }

Functions

- **LengthUnit HepMC::Units::default_length_unit ()**
default unit is defined by configure
- **MomentumUnit HepMC::Units::default_momentum_unit ()**
default unit is defined by configure
- **std::string HepMC::Units::name (MomentumUnit)**
convert enum to string
- **std::string HepMC::Units::name (LengthUnit)**
convert enum to string
- **double HepMC::Units::conversion_factor (MomentumUnit from, MomentumUnit to)**
scaling factor relative to MeV
- **double HepMC::Units::conversion_factor (LengthUnit from, LengthUnit to)**

9.79 VectorConversion.h File Reference

```
#include "HepMC/SimpleVector.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Namespaces

- namespace **CLHEP**

Functions

- **CLHEP::Hep3Vector convertTo (const HepMC::ThreeVector &v)**
Convert from HepMC::ThreeVector (p.238) to CLHEP::Hep3Vector.
- **CLHEP::HepLorentzVector convertTo (const HepMC::FourVector &v)**
Convert from HepMC::FourVector (p.50) to CLHEP::HepLorentzVector.

9.79.1 Function Documentation

9.79.1.1 CLHEP::HepLorentzVector convertTo (const HepMC::FourVector &v) [inline]

Convert from **HepMC::FourVector** (p.50) to **CLHEP::HepLorentzVector**.
Definition at line 25 of file `VectorConversion.h`.
References `v`.

9.79.1.2 CLHEP::Hep3Vector convertTo (const HepMC::ThreeVector &v) [inline]

Convert from **HepMC::ThreeVector** (p.238) to **CLHEP::Hep3Vector**.

Examples:

example_BuildEventFromScratch.cc, and **VectorConversion.h**.

Definition at line 21 of file `VectorConversion.h`.
References `v`.
Referenced by `main()`.

9.80 Version.h File Reference

```
#include <string>
#include <iostream>
#include "HepMC/HepMCDefs.h"
```

Namespaces

- namespace **HepMC**

Functions

- **void HepMC::version ()**
print HepMC (p. 19) version
- **void HepMC::writeVersion (std::ostream &os)**
write HepMC (p. 19) version to os
- **std::string HepMC::versionName ()**
return HepMC (p. 19) version

9.81 WeightContainer.h File Reference

```
#include <iostream>
#include <vector>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::WeightContainer**
Container for the Weights associated with an event or vertex.

Chapter 10

HepMC Example Documentation

10.1 example_BuildEventFromScratch.cc

Example of building an event and a particle data table from scratch
This is meant to be of use for persons implementing **HepMC** (p.19) inside
a MC event generator

```
1
2 // Matt.Dobbs@Cern.CH, Feb 2000
3 // Example of building an event and a particle data table from scratch
4 // This is meant to be of use for persons implementing HepMC inside a MC
5 // event generator
6 // To Compile: go to the HepMC directory and type:
7 // gmake examples/example_BuildEventFromScratch.exe
8 //
9
10
11 #include <iostream>
12
13 #include "VectorConversion.h"
14 #include "HepMC/GenEvent.h"
15 #include "HepMC/ParticleDataTable.h"
16 #include "CLHEP/Vector/LorentzVector.h"
17
18 // in this example we use the HepMC namespace, so that we do not have to
19 // precede all HepMC classes with HepMC::
20
21 // This example also shows how to use the CLHEP Lorentz vector with HepMC2
22
23 using namespace HepMC;
24 using namespace CLHEP;
25
26 int main() {
27     //
28     // In this example we will place the following event into HepMC "by hand"
29     //
30     //      name status pdg_id  parent Px      Py      Pz      Energy      Mass
31     //  1  !p+!      3    2212    0,0    0.000    0.000 7000.000 7000.000    0.938
32     //  2  !p+!      3    2212    0,0    0.000    0.000-7000.000 7000.000    0.938
33     //=====
34     //  3  !d!       3        1    1,1    0.750   -1.569   32.191   32.238    0.000
35     //  4  !u~!      3       -2    2,2   -3.047  -19.000  -54.629   57.920    0.000
36     //  5  !W-!      3      -24    1,2    1.517   -20.68   -20.605   85.925   80.799
37     //  6  !gamma!   1        2    1,2   -3.813    0.113   -1.833    4.233    0.000
38     //  7  !d!       1         1    5,5   -2.445   28.816    6.082   29.552    0.010
39     //  8  !u~!      1       -2    5,5    3.962  -49.498  -26.687   56.373    0.006
40 }
```

```

41 // first we construct a ParticleDataTable with all the particles we need
42 ParticleDataTable pdt("my particle data table");
43 // create a particle data entry for the proton and add it to pdt at the
44 // same time
45 pdt.insert( new ParticleData( "p+", 2212, +1, 0.938, -1, .5 ) );
46 pdt.insert( new ParticleData( "d", 1, -2./3., 0, -1, .5 ) );
47 pdt.insert( new ParticleData( "u~", -2, -1./3., 0, -1, .5 ) );
48 pdt.insert( new ParticleData( "W-", -24, -1, 80.396,
49                               clifetime_from_width(2.06), 1 ) );
50 pdt.insert( new ParticleData( "gamma", 22, 0, 0, -1, 1 ) );
51
52 // print out the GenParticle Data to the screen
53 pdt.print();
54
55 // now we build the graph, which will look like
56 //
57 // p1                                     #
58 // \v1__p3      p5---v4                #
59 //      \_v3_/      \                  #
60 //      /      \      p8                #
61 //      v2__p4      \                  #
62 //      /      p6                  #
63 // p2                                     #
64 //                                     #
65
66 // First create the event container, with Signal Process 20, event number 1
67 //
68 // Note that the HepLorentzVectors will be automatically converted to
69 // HepMC::FourVector within GenParticle and GenVertex
70 GenEvent* evt = new GenEvent( 20, 1 );
71 // define the units
72 evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
73 //
74 // create vertex 1 and vertex 2, together with their inparticles
75 GenVertex* v1 = new GenVertex();
76 evt->add_vertex( v1 );
77 v1->add_particle_in( new GenParticle( HepLorentzVector(0,0,7000,7000),
78                                     2212, 3 ) );
79 GenVertex* v2 = new GenVertex();
80 evt->add_vertex( v2 );
81 v2->add_particle_in( new GenParticle( HepLorentzVector(0,0,-7000,7000),
82                                     2212, 3 ) );
83 //
84 // create the outgoing particles of v1 and v2
85 GenParticle* p3 =
86     new GenParticle( HepLorentzVector(.750,-1.569,32.191,32.238), 1, 3 );
87 v1->add_particle_out( p3 );
88 GenParticle* p4 =
89     new GenParticle( HepLorentzVector(-3.047,-19.,-54.629,57.920), -2, 3 );
90 v2->add_particle_out( p4 );
91 //
92 // create v3
93 GenVertex* v3 = new GenVertex();
94 evt->add_vertex( v3 );
95 v3->add_particle_in( p3 );
96 v3->add_particle_in( p4 );
97 v3->add_particle_out(
98     new GenParticle( HepLorentzVector(-3.813,0.113,-1.833,4.233 ), 22, 1 )
99 );
100 GenParticle* p5 =
101     new GenParticle( HepLorentzVector(1.517,-20.68,-20.605,85.925), -24,3);
102 v3->add_particle_out( p5 );
103 //
104 // create v4
105 GenVertex* v4 = new GenVertex(HepLorentzVector(0.12,-0.3,0.05,0.004));
106 evt->add_vertex( v4 );
107 v4->add_particle_in( p5 );

```



```
108     v4->add_particle_out(
109         new GenParticle( HepLorentzVector(-2.445,28.816,6.082,29.552), 1,1 )
110     );
111     v4->add_particle_out(
112         new GenParticle( HepLorentzVector(3.962,-49.498,-26.687,56.373), -2,1 )
113     );
114     //
115     // tell the event which vertex is the signal process vertex
116     evt->set_signal_process_vertex( v3 );
117     // the event is complete, we now print it out to the screen
118     evt->print();
119
120     // example conversion back to Lorentz vector
121     // add all outgoing momenta
122     std::cout << std::endl;
123     std::cout << " Add output momenta " << std::endl;
124     HepLorentzVector sum;
125     for ( GenEvent::particle_const_iterator p = evt->particles_begin();
126         p != evt->particles_end(); ++p ){
127         if( (*p)->status() == 1 ) {
128             sum += convertTo( (*p)->momentum() );
129             (*p)->print();
130         }
131     }
132     std::cout << "Vector Sum: " << sum << std::endl;
133
134     // now clean-up by deleteing all objects from memory
135     //
136     // deleting the event deletes all contained vertices, and all particles
137     // contained in those vertices
138     delete evt;
139
140     // delete all particle data objects in the particle data table pdt
141     pdt.delete_all();
142
143     return 0;
144 }
```

10.2 example_EventSelection.cc

Example of applying an event selection to the events written to file using example_MyPythia.cxx Events containing a photon of $p_T > 25$ GeV pass the selection and are written to "example_EventSelection.dat"

```

1
2 // Matt.Dobbs@Cern.CH, Feb 2000
3 // Example of applying an event selection to the events written to file
4 // using example_MyPythia.cxx
5 // Events containing a photon of  $p_T > 25$  GeV pass the selection and are
6 // written to "example_EventSelection.dat"
7 // To Compile: go to the HepMC directory and type:
8 // gmake examples/example_EventSelection.exe
9 //
10 //
11
12 #include "HepMC/IO_GenEvent.h"
13 #include "HepMC/GenEvent.h"
14
15
16
17 class IsEventGood {
18 public:
19     bool operator()( const HepMC::GenEvent* evt ) {
20         for ( HepMC::GenEvent::particle_const_iterator p
21              = evt->particles_begin(); p != evt->particles_end(); ++p ){
22             if ( (*p)->pdg_id() == 22 && (*p)->momentum().perp() > 25. ) {
23                 //std::cout << "Event " << evt->event_number()
24                 //      << " is a good event." << std::endl;
25                 //(*p)->print();
26                 return 1;
27             }
28         }
29         return 0;
30     }
31 };
32
33 int main() {
34     // declare an input strategy to read the data produced with the
35     // example_MyPythia
36     { // begin scope of ascii_in and ascii_out
37         HepMC::IO_GenEvent ascii_in("example_MyPythia.dat",std::ios::in);
38         // declare another IO_GenEvent for writing out the good events
39         HepMC::IO_GenEvent ascii_out("example_EventSelection.dat",std::ios::out);
40         // declare an instance of the event selection predicate
41         IsEventGood is_good_event;
42         //.....EVENT LOOP
43         int icount=0;
44         int num_good_events=0;
45         HepMC::GenEvent* evt = ascii_in.read_next_event();
46         while ( evt ) {
47             icount++;
48             if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
49                                     << " its # " << evt->event_number()
50                                     << std::endl;
51             if ( is_good_event(evt) ) {
52                 ascii_out << evt;
53                 ++num_good_events;
54             }
55             delete evt;
56             ascii_in >> evt;
57         }
58         //.....PRINT RESULT
59         std::cout << num_good_events << " out of " << icount
60                 << " processed events passed the cuts. Finished." << std::endl;
61     } // end scope of ascii_in and ascii_out
62     return 0;
63 }

```

```
67 }  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77
```

10.3 example_MyHerwig.cc

```

1
2 // Matt.Dobbs@Cern.CH, October 2002
3 // example of generating events with Herwig using HepMC/HerwigWrapper.h
4 // Events are read into the HepMC event record from the FORTRAN HEPEVT
5 // common block using the IO_HERWIG strategy.
6
7 #include <iostream>
8 #include "HepMC/HerwigWrapper.h"
9 #include "HepMC/IO_HERWIG.h"
10 #include "HepMC/IO_GenEvent.h"
11 #include "HepMC/GenEvent.h"
12 #include "HepMC/HEPEVT_Wrapper.h"
13 #include "HerwigHelper.h"
14
15 int main() {
16     //
17     //.....HEPEVT
18     // Herwig 6.4 uses HEPEVT with 4000 entries and 8-byte floating point
19     // numbers. We need to explicitly pass this information to the
20     // HEPEVT_Wrapper.
21     //
22     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
23     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
24     //
25     //.....INITIALIZATIONS
26
27     hwproc.PBEAM1 = 7000.; // energy of beam1
28     hwproc.PBEAM2 = 7000.; // energy of beam2
29     // 1610 = gg->H--> WW, 1706 = qq-->ttbar, 2510 = ttH -> ttWW
30     hwproc.IPROC = 1706; // qq -> ttbar production
31     hwproc.MAXEV = 100; // number of events
32     // tell it what the beam particles are:
33     for ( unsigned int i = 0; i < 8; ++i ) {
34         hwbmch.PART1[i] = (i < 1) ? 'P' : ' ';
35         hwbmch.PART2[i] = (i < 1) ? 'P' : ' ';
36     }
37     hwigin(); // INITIALISE OTHER COMMON BLOCKS
38     hwevnt.MAXPR = 1; // number of events to print
39     hwiinc(); // compute parameter-dependent constants
40     hweini(); // initialise elementary process
41
42     //.....HepMC INITIALIZATIONS
43     //
44     // Instantiate an IO strategy for reading from HEPEVT.
45     HepMC::IO_HERWIG hepevtio;
46     // Instantiate an IO strategy to write the data to file
47     HepMC::IO_GenEvent ascii_io("example_MyHerwig.dat",std::ios::out);
48     //
49     //.....EVENT LOOP
50     for ( int i = 1; i <= hwproc.MAXEV; i++ ) {
51         if ( i%50==1 ) std::cout << "Processing Event Number "
52             << i << std::endl;
53
54         // initialise event
55         hwiue();
56         // generate hard subprocess
57         hwepro();
58         // generate parton cascades
59         hwbgen();
60         // do heavy object decays
61         hwdhob();
62         // do cluster formation
63         hwcfor();
64         // do cluster decays
65         hwcdec();
66         // do unstable particle decays

```

```
76     hwdhad();
77     // do heavy flavour hadron decays
78     hwdhvy();
79     // add soft underlying event if needed
80     hwmevt();
81     // finish event
82     hwufne();
83     HepMC::GenEvent* evt = hepevtio.read_next_event();
84     // define the units (Herwig uses GeV and mm)
85     evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
86     // set cross section information
87     evt->set_cross_section( getHerwigCrossSection(i) );
88     // add some information to the event
89     evt->set_event_number(i);
90     evt->set_signal_process_id(20);
91     if (i<=hwevnt.MAXPR) {
92         std::cout << "\n\n This is the FIXED version of HEPEVT as "
93             << "coded in IO_HERWIG " << std::endl;
94         HepMC::HEPEVT_Wrapper::print_hepevt();
95         evt->print();
96     }
97     // write the event to the ascii file
98     ascii_io << evt;
99
100     // we also need to delete the created event from memory
101     delete evt;
102 }
103 //.....TERMINATION
104 hwefin();
105
106 return 0;
107 }
```

10.4 example_MyPythia.cc

example to generate events and write output example to generate events
and perform simple event selection example to read the file written by
pythia_out example to generate events, write them, and read them back

```

1
2 // Matt.Dobbs@Cern.CH, December 1999
3 // November 2000, updated to use Pythia 6.1
4 //
46
47
48 #include <iostream>
49 #include "HepMC/PythiaWrapper.h"
50 #include "HepMC/IO_HEPEVT.h"
51 #include "HepMC/IO_GenEvent.h"
52 #include "HepMC/IO_AsciiParticles.h"
53 #include "HepMC/GenEvent.h"
54 #include "PythiaHelper.h"
55
56
57
61 class IsGoodEventMyPythia {
62 public:
63     bool operator()( const HepMC::GenEvent* evt ) {
64         for ( HepMC::GenEvent::particle_const_iterator p
65              = evt->particles_begin(); p != evt->particles_end(); ++p ){
66             if ( (*p)->pdg_id() == 22 && (*p)->momentum().perp() > 25. ) {
67                 //std::cout << "Event " << evt->event_number()
68                 //      << " is a good event." << std::endl;
69                 //(*p)->print();
70                 return 1;
71             }
72         }
73         return 0;
74     }
75 };
76
77
78
79 void pythia_out();
80 void pythia_in();
81 void pythia_in_out();
82 void event_selection();
83 void pythia_particle_out();
84
85 int main() {
86     // example to generate events and write output
87     pythia_out();
88     // example to generate events and perform simple event selection
89     event_selection();
90     // example to read the file written by pythia_out
91     pythia_in();
92     // example to generate events, write them, and read them back
93     pythia_in_out();
94
95     return 0;
96 }
97
98
99 void pythia_out()
100 {
101     std::cout << std::endl;
102     std::cout << "Begin pythia_out()" << std::endl;
103     //.....HEPEVT
104     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
105     // numbers. We need to explicitly pass this information to the
106     // HEPEVT_Wrapper.

```

```

107  //
108  HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
109  HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
110  //
111  //.....PYTHIA INITIALIZATIONS
112  initPythia();
113
114  //.....HepMC INITIALIZATIONS
115  //
116  // Instantiate an IO strategy for reading from HEPEVT.
117  HepMC::IO_HEPEVT hepevtio;
118  //
119  { // begin scope of ascii_io
120      // Instantiate an IO strategy to write the data to file
121      HepMC::IO_GenEvent ascii_io("example_MyPythia.dat",std::ios::out);
122      //
123      //.....EVENT LOOP
124      for ( int i = 1; i <= 100; i++ ) {
125          if ( i%50==1 ) std::cout << "Processing Event Number "
126                                  << i << std::endl;
127          call_pyevnt(); // generate one event with Pythia
128          // pythia pyhepc routine converts common PYJETTS in common HEPEVT
129          call_pyhepc( 1 );
130          HepMC::GenEvent* evt = hepevtio.read_next_event();
131          // define the units (Pythia uses GeV and mm)
132          evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
133          // add some information to the event
134          evt->set_event_number(i);
135          evt->set_signal_process_id(20);
136          // set number of multi parton interactions
137          evt->set_mpi( pypars.msti[31-1] );
138          // set cross section information
139          evt->set_cross_section( getPythiaCrossSection() );
140          // write the event out to the ascii files
141          ascii_io << evt;
142          // we also need to delete the created event from memory
143          delete evt;
144      }
145      //.....TERMINATION
146      // write out some information from Pythia to the screen
147      call_pystat( 1 );
148  } // end scope of ascii_io
149 }
150
151
152 void event_selection()
153 {
154     std::cout << std::endl;
155     std::cout << "Begin event_selection()" << std::endl;
156     //.....HEPEVT
157     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
158     // numbers. We need to explicitly pass this information to the
159     // HEPEVT_Wrapper.
160     //
161     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
162     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
163     //
164     //.....PYTHIA INITIALIZATIONS
165     initPythia();
166     //
167     //.....HepMC INITIALIZATIONS
168     // Instantiate an IO strategy for reading from HEPEVT.
169     HepMC::IO_HEPEVT hepevtio;
170     // declare an instance of the event selection predicate
171     IsGoodEventMyPythia is_good_event;
172     //.....EVENT LOOP
173     int icount=0;

```

```

174     int num_good_events=0;
175     for ( int i = 1; i <= 100; i++ ) {
176         icount++;
177         if ( i%50==1 ) std::cout << "Processing Event Number "
178                                 << i << std::endl;
179         call_pyevnt(); // generate one event with Pythia
180         // pythia pyhepc routine convert common PYJETS in common HEPEVT
181         call_pyhepc( 1 );
182         HepMC::GenEvent* evt = hepevtio.read_next_event();
183         // define the units (Pythia uses GeV and mm)
184         evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
185         // set number of multi parton interactions
186         evt->set_mpi( pypars.msti[31-1] );
187         // set cross section information
188         evt->set_cross_section( getPythiaCrossSection() );
189         // do event selection
190         if ( is_good_event(evt) ) {
191             std::cout << "Good Event Number " << i << std::endl;
192             ++num_good_events;
193         }
194         // we also need to delete the created event from memory
195         delete evt;
196     }
197     //.....TERMINATION
198     // write out some information from Pythia to the screen
199     call_pystat( 1 );
200     //.....PRINT RESULTS
201     std::cout << num_good_events << " out of " << icount
202             << " processed events passed the cuts. Finished." << std::endl;
203 }
204
205 void pythia_in()
206 {
207     std::cout << std::endl;
208     std::cout << "Begin pythia_in()" << std::endl;
209     std::cout << "reading example_MyPythia.dat" << std::endl;
210     //.....define an input scope
211     {
212         // open input stream
213         std::ifstream istr( "example_MyPythia.dat" );
214         if( !istr ) {
215             std::cerr << "example_ReadMyPythia: cannot open example_MyPythia.dat" << std::endl;
216             exit(-1);
217         }
218         HepMC::IO_GenEvent ascii_in(istr);
219         // open output stream (alternate method)
220         HepMC::IO_GenEvent ascii_out("example_MyPythia2.dat",std::ios::out);
221         // now read the file
222         int icount=0;
223         HepMC::GenEvent* evt = ascii_in.read_next_event();
224         while ( evt ) {
225             icount++;
226             if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
227                                     << " its # " << evt->event_number()
228                                     << std::endl;
229             // write the event out to the ascii file
230             ascii_out << evt;
231             delete evt;
232             ascii_in >> evt;
233         }
234         //.....PRINT RESULT
235         std::cout << icount << " events found. Finished." << std::endl;
236     } // ascii_out and istr destructors are called here
237 }
238
239 void pythia_in_out()
240 {

```



```

241     std::cout << std::endl;
242     std::cout << "Begin pythia_in_out()" << std::endl;
243     //.....HEPEVT
244     // Pythia 6.3 uses HEPEVT with 4000 entries and 8-byte floating point
245     // numbers. We need to explicitly pass this information to the
246     // HEPEVT_Wrapper.
247     //
248     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
249     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
250     //
251     //.....PYTHIA INITIALIZATIONS
252     initPythia();
253
254     //.....HepMC INITIALIZATIONS
255     //
256     // Instantiate an IO strategy for reading from HEPEVT.
257     HepMC::IO_HEPEVT hepevtio;
258     //
259     //.....define the output scope
260     {
261         // Instantial an IO strategy to write the data to file - it uses the
262         // same ParticleDataTable
263         HepMC::IO_GenEvent ascii_io("example_MyPythiaRead.dat",std::ios::out);
264         //
265         //.....EVENT LOOP
266         for ( int i = 1; i <= 100; i++ ) {
267             if ( i%50==1 ) std::cout << "Processing Event Number "
268                                     << i << std::endl;
269             call_pyevnt(); // generate one event with Pythia
270             // pythia pyhepc routine converts common PYJETS in common HEPEVT
271             call_pyhepc( 1 );
272             HepMC::GenEvent* evt = hepevtio.read_next_event();
273             // define the units (Pythia uses GeV and mm)
274             evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
275             // set cross section information
276             evt->set_cross_section( getPythiaCrossSection() );
277             // add some information to the event
278             evt->set_event_number(i);
279             evt->set_signal_process_id(20);
280             // write the event out to the ascii file
281             ascii_io << evt;
282             // we also need to delete the created event from memory
283             delete evt;
284         }
285         //.....TERMINATION
286         // write out some information from Pythia to the screen
287         call_pystat( 1 );
288     } // ascii_io destructor is called here
289     //
290     //.....define an input scope
291     {
292         // now read the file we wrote
293         HepMC::IO_GenEvent ascii_in("example_MyPythiaRead.dat",std::ios::in);
294         HepMC::IO_GenEvent ascii_io2("example_MyPythiaRead2.dat",std::ios::out);
295         int icount=0;
296         HepMC::GenEvent* evt = ascii_in.read_next_event();
297         while ( evt ) {
298             icount++;
299             if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
300                                     << " its # " << evt->event_number()
301                                     << std::endl;
302             // write the event out to the ascii file
303             ascii_io2 << evt;
304             delete evt;
305             ascii_in >> evt;
306         }
307         //.....PRINT RESULT

```

```

308         std::cout << icount << " events found. Finished." << std::endl;
309     } // ascii_io2 and ascii_in destructors are called here
310 }
311
312 void pythia_particle_out()
313 {
314     std::cout << std::endl;
315     std::cout << "Begin pythia_particle_out()" << std::endl;
316     //.....HEPEVT
317     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
318     // numbers. We need to explicitly pass this information to the
319     // HEPEVT_Wrapper.
320     //
321     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
322     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
323     //
324     //.....PYTHIA INITIALIZATIONS
325     initPythia();
326
327     //.....HepMC INITIALIZATIONS
328     //
329     // Instantiate an IO strategy for reading from HEPEVT.
330     HepMC::IO_HEPEVT hepevtio;
331     //
332     { // begin scope of ascii_io
333         // Instantiate an IO strategy to write the data to file
334         HepMC::IO_AsciiParticles ascii_io("example_PythiaParticle.dat",std::ios::out);
335         //
336         //.....EVENT LOOP
337         for ( int i = 1; i <= 100; i++ ) {
338             if ( i%50==1 ) std::cout << "Processing Event Number "
339                                     << i << std::endl;
340             call_pyevnt(); // generate one event with Pythia
341             // pythia pyhepc routine converts common PYJETS in common HEPEVT
342             call_pyhepc( 1 );
343             HepMC::GenEvent* evt = hepevtio.read_next_event();
344             // define the units (Pythia uses GeV and mm)
345             evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
346             // set cross section information
347             evt->set_cross_section( getPythiaCrossSection() );
348             // add some information to the event
349             evt->set_event_number(i);
350             evt->set_signal_process_id(20);
351             // write the event out to the ascii file
352             ascii_io << evt;
353             // we also need to delete the created event from memory
354             delete evt;
355         }
356         //.....TERMINATION
357         // write out some information from Pythia to the screen
358         call_pystat( 1 );
359     } // end scope of ascii_io
360 }
361

```

10.5 example_MyPythiaOnlyToHepMC.cc

Example of generating events with Pythia using **HepMC/PythiaWrapper.h** (p.327) Events are read into the **HepMC** (p.19) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy - nothing is done with them. This program is just used to find the total time required to transfer from HEPEVT into the **HepMC** (p.19) event record.

```

1
2 // Matt.Dobbs@Cern.CH, December 1999
3 // November 2000, updated to use Pythia 6.1
4 // example of generating events with Pythia
5 // using HepMC/PythiaWrapper.h
6 // Events are read into the HepMC event record from the FORTRAN HEPEVT
7 // common block using the IO_HEPEVT strategy -- nothing is done with them.
8 // This program is just used to find the total time required to transfer
9 // from HEPEVT into the HepMC event record.
11 // To Compile: go to the HepMC directory and type:
12 // gmake examples/example_MyPythiaOnlyTo HepMC.exe
13 //
14 // See comments in examples/example_MyPythia.cxx regarding the HEPEVT wrapper.
15 //
16
17 #include <iostream>
18 #include "HepMC/PythiaWrapper.h"
19 #include "HepMC/IO_HEPEVT.h"
20 #include "HepMC/GenEvent.h"
21 #include "PythiaHelper.h"
22
23 int main() {
24     //
25     //.....HEPEVT
26     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
27     // numbers. We need to explicitly pass this information to the
28     // HEPEVT_Wrapper.
29     //
30     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
31     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
32     //
33     //.....PYTHIA INITIALIZATIONS
34     initPythia();
35     //
36     //.....HepMC INITIALIZATIONS
37     //
38     // Instantiate an IO strategy for reading from HEPEVT.
39     HepMC::IO_HEPEVT hepevtio;
40     //
41     //.....EVENT LOOP
42     for ( int i = 1; i <= 100; i++ ) {
43         if ( i%50==1 ) std::cout << "Processing Event Number "
44             << i << std::endl;
45         call_pyevnt(); // generate one event with Pythia
46         // pythia pyhepc routine convert common PYJETS in common HEPEVT
47         call_pyhepc( 1 );
48         HepMC::GenEvent* evt = hepevtio.read_next_event();
49         // define the units (Pythia uses GeV and mm)
50         evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
51         // set number of multi parton interactions
52         evt->set_mpi( pypars.msti[31-1] );
53         // set cross section information
54         evt->set_cross_section( getPythiaCrossSection() );
55         //
56         //.....USER WOULD PROCESS EVENT HERE
57         //
58         // we also need to delete the created event from memory

```

```
59         delete evt;
60     }
61     //.....TERMINATION
62     // write out some information from Pythia to the screen
63     call_pystat( 1 );
64
65     return 0;
66 }
67
68
69
```

10.6 example_PythiaStreamIO.cc

This example generates Pythia events and fills cross section information from pyint5. The example uses streaming I/O to write a file and then read it.

```

1
2 // example_PythiaStreamIO.cc
3 //
4 // garren@fnal.gov, May 2009
5 //
19
20
21 #include <fstream>
22 #include <iostream>
23 #include "HepMC/PythiaWrapper.h"
24 #include "HepMC/IO_HEPEVT.h"
25 #include "HepMC/GenEvent.h"
26 #include "PythiaHelper.h"
27
28 void writePythiaStreamIO();
29 void readPythiaStreamIO();
30
31 int main() {
32
33     writePythiaStreamIO();
34     readPythiaStreamIO();
35
36     return 0;
37 }
38
39
40 void writePythiaStreamIO() {
41     // example to generate events and write output
42     std::cout << std::endl;
43     std::cout << "Begin pythia_out()" << std::endl;
44     //.....HEPEVT
45     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
46     // numbers. We need to explicitly pass this information to the
47     // HEPEVT_Wrapper.
48     //
49     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
50     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
51     //
52     //.....PYTHIA INITIALIZATIONS
53     initPythia();
54
55     //.....HepMC INITIALIZATIONS
56     //
57     // Instantiate an IO strategy for reading from HEPEVT.
58     HepMC::IO_HEPEVT hepevtio;
59     //
60     { // begin scope of ascii_io
61         // declare an output stream
62         const char outfile[] = "example_PythiaStreamIO_write.dat";
63         std::ofstream ascii_io( outfile );
64         if( !ascii_io ) {
65             std::cerr << "cannot open " << outfile << std::endl;
66             exit(-1);
67         }
68         // use the default IO_GenEvent precision
69         ascii_io.precision(16);
70         // write the line that defines the beginning of a GenEvent block
71         HepMC::write_HepMC_IO_block_begin( ascii_io );
72         //
73         //.....EVENT LOOP

```

```

74     for ( int i = 1; i <= 100; i++ ) {
75         if ( i%50==1 ) std::cout << "Processing Event Number "
76                                 << i << std::endl;
77         call_pyevnt();           // generate one event with Pythia
78         // pythia pyhepc routine converts common PYJETS in common HEPEVT
79         call_pyhepc( 1 );
80         HepMC::GenEvent* evt = hepevtio.read_next_event();
81         // define the units (Pythia uses GeV and mm)
82         evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
83         // add some information to the event
84         evt->set_event_number(i);
85         evt->set_signal_process_id(20);
86         // set number of multi parton interactions
87         evt->set_mpi( pypars.msti[31-1] );
88         // set cross section information
89         evt->set_cross_section( getPythiaCrossSection() );
90         // write the event out to the ascii files
91         ascii_io << (*evt);;
92         // we also need to delete the created event from memory
93         delete evt;
94     }
95     // write the line that defines the end of a GenEvent block
96     HepMC::write_HepMC_IO_block_end( ascii_io );
97     //.....TERMINATION
98     // write out some information from Pythia to the screen
99     call_pystat( 1 );
100 } // end scope of ascii_io
101 }
102
103 void readPythiaStreamIO() {
104     // example to read events written by writePythiaStreamIO
105     // and write them back out
106     std::cout << std::endl;
107     // input units are GeV and mm
108     const char infile[] = "example_PythiaStreamIO_write.dat";
109     std::ifstream is( infile );
110     if( !is ) {
111         std::cerr << "cannot open " << infile << std::endl;
112         exit(-1);
113     }
114     //
115     { // begin scope of ascii_io
116         // declare an output stream
117         const char outfile[] = "example_PythiaStreamIO_read.dat";
118         std::ofstream ascii_io( outfile );
119         if( !ascii_io ) {
120             std::cerr << "cannot open " << outfile << std::endl;
121             exit(-1);
122         }
123         ascii_io.precision(16);
124         HepMC::write_HepMC_IO_block_begin( ascii_io );
125         //
126         //.....EVENT LOOP
127         HepMC::GenEvent evt;
128         int i = 0;
129         while ( is ) {
130             evt.read( is );
131             // make sure we have a valid event
132             if( evt.is_valid() ) {
133                 ++i;
134                 if ( i%50==1 ) std::cout << "Processing Event Number "
135                                         << i << std::endl;
136                 if ( i%25==2 ) {
137                     // write the cross section if it exists
138                     if( evt.cross_section() ) {
139                         std::cout << "cross section at event " << i << " is "
140                                     << evt.cross_section()->cross_section()

```

```
141                                     << std::endl;
142                                     }
143                                     }
144                                     // write the event out to the ascii files
145                                     evt.write( ascii_io );
146                                     }
147                                     }
148                                     //.....TERMINATION
149                                     HepMC::write_HepMC_IO_block_end( ascii_io );
150     } // end scope of ascii_io
151 }
```

10.7 example_UsingIterators.cc

This example shows how to use the particle and vertex iterators

```

1
2 // Matt.Dobbs@Cern.CH, Feb 2000
3 // This example shows how to use the particle and vertex iterators
5 // To Compile: go to the HepMC directory and type:
6 // gmake examples/example_UsingIterators.exe
7 //
8
9 #include "HepMC/IO_GenEvent.h"
10 #include "HepMC/GenEvent.h"
11 #include <math.h>
12 #include <algorithm>
13 #include <list>
14
15
16
17 class IsPhoton {
18 public:
19     bool operator()( const HepMC::GenParticle* p ) {
20         if ( p->pdg_id() == 22
21             && p->momentum().perp() > 10. ) return 1;
22         return 0;
23     }
24 };
25
26
27 class IsW_Boson {
28 public:
29     bool operator()( const HepMC::GenParticle* p ) {
30         if ( abs(p->pdg_id()) == 24 ) return 1;
31         return 0;
32     }
33 };
34
35
36 class IsStateFinal {
37 public:
38     bool operator()( const HepMC::GenParticle* p ) {
39         if ( !p->end_vertex() && p->status()==1 ) return 1;
40         return 0;
41     }
42 };
43
44
45 int main() {
46     { // begin scope of ascii_in
47         // an event has been prepared in advance for this example, read it
48         // into memory using the IO_GenEvent input strategy
49         HepMC::IO_GenEvent ascii_in("example_UsingIterators.txt",std::ios::in);
50         if ( ascii_in.rdstate() == std::ios::failbit ) {
51             std::cerr << "ERROR input file example_UsingIterators.txt is needed "
52                 << "and does not exist. "
53                 << "\n Look for it in HepMC/examples, Exit." << std::endl;
54             return 1;
55         }
56
57         HepMC::GenEvent* evt = ascii_in.read_next_event();
58
59         // if you wish to have a look at the event, then use evt->print();
60
61         // use GenEvent::vertex_iterator to fill a list of all
62         // vertices in the event
63         std::list<HepMC::GenVertex*> allvertices;
64         for ( HepMC::GenEvent::vertex_iterator v = evt->vertices_begin();
65              v != evt->vertices_end(); ++v ) {

```



```

77         allvertices.push_back(*v);
78     }
79
80     // we could do the same thing with the STL algorithm copy
81     std::list<HepMC::GenVertex*> allvertices2;
82     copy( evt->vertices_begin(), evt->vertices_end(),
83           back_inserter(allvertices2) );
84
85     // fill a list of all final state particles in the event, by requiring
86     // that each particle satisfyies the IsStateFinal predicate
87     IsStateFinal isfinal;
88     std::list<HepMC::GenParticle*> finalstateparticles;
89     for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
90           p != evt->particles_end(); ++p ) {
91         if ( isfinal(*p) ) finalstateparticles.push_back(*p);
92     }
93
94     // an STL-like algorithm called HepMC::copy_if is provided in the
95     // GenEvent.h header to do this sort of operation more easily,
96     // you could get the identical results as above by using:
97     std::list<HepMC::GenParticle*> finalstateparticles2;
98     HepMC::copy_if( evt->particles_begin(), evt->particles_end(),
99                     back_inserter(finalstateparticles2), IsStateFinal() );
100
101     // lets print all photons in the event that satisfy the IsPhoton criteria
102     IsPhoton isphoton;
103     for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
104           p != evt->particles_end(); ++p ) {
105         if ( isphoton(*p) ) (*p)->print();
106     }
107
108     // the GenVertex::particle_iterator and GenVertex::vertex_iterator
109     // are slightly different from the GenEvent:: versions, in that
110     // the iterator starts at the given vertex, and walks through the attached
111     // vertex returning particles/vertices.
112     // Thus only particles/vertices which are in the same graph as the given
113     // vertex will be returned. A range is specified with these iterators,
114     // the choices are:
115     //   parents, children, family, ancestors, descendants, relatives
116     // here are some examples.
117
118     // use GenEvent::particle_iterator to find all W's in the event,
119     // then
120     // (1) for each W user the GenVertex::particle_iterator with a range of
121     //     parents to return and print the immediate mothers of these W's.
122     // (2) for each W user the GenVertex::particle_iterator with a range of
123     //     descendants to return and print all descendants of these W's.
124     IsW_Boson isw;
125     for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
126           p != evt->particles_end(); ++p ) {
127         if ( isw(*p) ) {
128             std::cout << "A W boson has been found: " << std::endl;
129             (*p)->print();
130             // return all parents
131             // we do this by pointing to the production vertex of the W
132             // particle and asking for all particle parents of that vertex
133             std::cout << "\t Its parents are: " << std::endl;
134             if ( (*p)->production_vertex() ) {
135                 for ( HepMC::GenVertex::particle_iterator mother
136                       = (*p)->production_vertex()->
137                         particles_begin(HepMC::parents);
138                       mother != (*p)->production_vertex()->
139                         particles_end(HepMC::parents);
140                       ++mother ) {
141                 std::cout << "\t";
142                 (*mother)->print();
143             }

```

```
144         }
145         // return all descendants
146         // we do this by pointing to the end vertex of the W
147         // particle and asking for all particle descendants of that vertex
148         std::cout << "\t\t Its descendants are: " << std::endl;
149         if ( (*p)->end_vertex() ) {
150             for ( HepMC::GenVertex::particle_iterator des
151                  = (*p)->end_vertex()->
152                    particles_begin(HepMC::descendants);
153                  des != (*p)->end_vertex()->
154                    particles_end(HepMC::descendants);
155                  ++des ) {
156                 std::cout << "\t\t";
157                 (*des)->print();
158             }
159         }
160     }
161 }
162 // cleanup
163 delete evt;
164 // in analogy to the above, similar use can be made of the
165 // HepMC::GenVertex::vertex_iterator, which also accepts a range.
166 } // end scope of ascii_in
167
168 return 0;
169 }
```

10.8 testFlow.cc

Use a modified example_BuildEventFromScratch to test Flow

```

1
2 // testFlow.cc
3 //
4 // garren@fnal.gov, June 2009
5 // based on example_BuildEventFromScratch.cc
6
7
8 #include <iostream>
9 #include <fstream>
10 #include <vector>
11
12 #include "HepMC/GenEvent.h"
13 #include "HepMC/IO_GenEvent.h"
14
15 typedef std::vector<HepMC::GenParticle*> FlowVec;
16
17 int main() {
18     //
19     // In this example we will place the following event into HepMC "by hand"
20     //
21     //      name status pdg_id  parent Px      Py      Pz      Energy      Mass
22     //  1  !p+!      3    2212    0,0    0.000    0.000 7000.000 7000.000    0.938
23     //  2  !p+!      3    2212    0,0    0.000    0.000-7000.000 7000.000    0.938
24     //=====
25     //  3  !d!        3        1    1,1    0.750   -1.569   32.191   32.238    0.000
26     //  4  !u~!       3       -2    2,2   -3.047  -19.000  -54.629   57.920    0.000
27     //  5  !W-!       3      -24    1,2    1.517   -20.68   -20.605   85.925   80.799
28     //  6  !gamma!    1        22    1,2   -3.813    0.113   -1.833    4.233    0.000
29     //  7  !d!        1         1    5,5   -2.445   28.816    6.082   29.552    0.010
30     //  8  !u~!       1       -2    5,5    3.962  -49.498  -26.687   56.373    0.006
31
32     // open an output file
33     const char outfile[] = "testFlow.out";
34     std::ofstream os( outfile );
35     if( !os ) {
36         std::cerr << "cannot open " << outfile << std::endl;
37         exit(-1);
38     }
39     // declare several IO_GenEvent instances for comparison
40     HepMC::IO_GenEvent xout1("testFlow.out1",std::ios::out);
41     HepMC::IO_GenEvent xout2("testFlow.out2",std::ios::out);
42     HepMC::IO_GenEvent xout3("testFlow.out3",std::ios::out);
43
44     int numbad = 0;
45
46
47     // build the graph, which will look like
48     //
49     //      p7
50     //      /
51     //  \v1__p3      p5---v4
52     //      \_v3_/
53     //      /      \
54     //  v2__p4      \
55     //      /      p6
56     //  p2
57     //
58     //      #
59     //      #
60     //      #
61     //      #
62     //      #
63     //      #
64
65     // define a flow pattern as  p1 -> p3 -> p6
66     //                          and p2 -> p4 -> p5
67
68     //
69
70     // First create the event container, with Signal Process 20, event number 1
71     //
72     HepMC::GenEvent* evt = new HepMC::GenEvent( 20, 1 );

```

```

64 //
65 // create vertex 1 and vertex 2, together with their inparticles
66 HepMC::GenVertex* v1 = new HepMC::GenVertex();
67 evt->add_vertex( v1 );
68 HepMC::GenParticle* p1 = new HepMC::GenParticle( HepMC::FourVector(0,0,7000,7000),
69                                                    2212, 3 );
70 p1->set_flow(1,231);
71 v1->add_particle_in( p1 );
72 HepMC::GenVertex* v2 = new HepMC::GenVertex();
73 evt->add_vertex( v2 );
74 HepMC::GenParticle* p2 = new HepMC::GenParticle( HepMC::FourVector(0,0,-7000,7000),
75                                                    2212, 3 );
76 p2->set_flow(1,243);
77 v2->add_particle_in( p2 );
78 //
79 // create the outgoing particles of v1 and v2
80 HepMC::GenParticle* p3 =
81     new HepMC::GenParticle( HepMC::FourVector(.750,-1.569,32.191,32.238),
82                             1, 3 );
83 p3->set_flow(1,231);
84 v1->add_particle_out( p3 );
85 HepMC::GenParticle* p4 =
86     new HepMC::GenParticle( HepMC::FourVector(-3.047,-19.,-54.629,57.920),
87                             -2, 3 );
88 p4->set_flow(1,243);
89 v2->add_particle_out( p4 );
90 //
91 // create v3
92 HepMC::GenVertex* v3 = new HepMC::GenVertex();
93 evt->add_vertex( v3 );
94 v3->add_particle_in( p3 );
95 v3->add_particle_in( p4 );
96 HepMC::GenParticle* p6 =
97     new HepMC::GenParticle( HepMC::FourVector(-3.813,0.113,-1.833,4.233 ),
98                             22, 1 );
99 p6->set_flow(1,231);
100 v3->add_particle_out( p6 );
101 HepMC::GenParticle* p5 =
102     new HepMC::GenParticle( HepMC::FourVector(1.517,-20.68,-20.605,85.925),
103                             -24, 3 );
104 p5->set_flow(1,243);
105 v3->add_particle_out( p5 );
106 //
107 // create v4
108 HepMC::GenVertex* v4 = new HepMC::GenVertex(HepMC::FourVector(0.12,-0.3,0.05,0.004));
109 evt->add_vertex( v4 );
110 v4->add_particle_in( p5 );
111 HepMC::GenParticle* p7 = new HepMC::GenParticle( HepMC::FourVector(-2.445,28.816,6.082,29.552), 1,
112 v4->add_particle_out( p7 );
113 HepMC::GenParticle* p8 = new HepMC::GenParticle( HepMC::FourVector(3.962,-49.498,-26.687,56.373),
114 v4->add_particle_out( p8 );
115 //
116 // tell the event which vertex is the signal process vertex
117 evt->set_signal_process_vertex( v3 );
118 // the event is complete, we now print it out
119 evt->print( os );
120
121 // look at the flow we created
122 os << std::endl;
123 FlowVec result1 = p1->flow().dangling_connected_partners( p1->flow().icode(1) );
124 FlowVec result2 = p1->flow().connected_partners( p1->flow().icode(1) );
125 FlowVec::iterator it;
126 os << "dangling partners of particle " << p1->barcode() << std::endl;
127 for( it = result1.begin(); it != result1.end(); ++it ) {
128     os << (*it)->barcode() << " ";
129     os.width(8);
130     os << (*it)->pdg_id() << " " << (*it)->flow(1) << std::endl;

```

```

131     }
132     os << "all partners of particle " << p1->barcode() << std::endl;
133     for( it = result2.begin(); it != result2.end(); ++it ) {
134         os << (*it)->barcode() << " " ;
135         os.width(8);
136         os << (*it)->pdg_id() << " " << (*it)->flow(1) << std::endl;
137     }
138     FlowVec result3 = p2->flow().dangling_connected_partners( p2->flow().icode(1) );
139     FlowVec result4 = p2->flow().connected_partners( p2->flow().icode(1) );
140     os << "dangling partners of particle " << p2->barcode() << std::endl;
141     for( it = result3.begin(); it != result3.end(); ++it ) {
142         os << (*it)->barcode() << " " ;
143         os.width(8);
144         os << (*it)->pdg_id() << " " << (*it)->flow(1) << std::endl;
145     }
146     os << "all partners of particle " << p2->barcode() << std::endl;
147     for( it = result4.begin(); it != result4.end(); ++it ) {
148         os << (*it)->barcode() << " " ;
149         os.width(8);
150         os << (*it)->pdg_id() << " " << (*it)->flow(1) << std::endl;
151     }
152     xout1 << evt;
153
154     // try changing and erasing flow
155     p2->set_flow(2,345);
156     xout2 << evt;
157     FlowVec result5 = p2->flow().connected_partners( p2->flow().icode(1) );
158     if ( result4 != result5 ) {
159         std::cerr << "ERROR: list of partners has changed after adding flow" << std::endl;
160         ++numbad;
161     }
162     // the flow method returns a copy,
163     // so we must set the flow again to change it
164     HepMC::Flow f2 = p2->flow();
165     if( f2.erase(2) ) {
166         p2->set_flow( f2 );
167     } else {
168         std::cerr << "ERROR: first erase was NOT successful" << std::endl;
169         ++numbad;
170     }
171     f2 = p2->flow();
172     if( f2.erase(2) ) {
173         std::cerr << "ERROR: second erase was successful" << std::endl;
174     }
175     xout3 << evt;
176     FlowVec result6 = p2->flow().connected_partners( p2->flow().icode(1) );
177     if ( result4 != result6 ) {
178         std::cerr << "ERROR: list of partners has changed after removing flow" << std::endl;
179         ++numbad;
180     }
181
182     // now clean-up by deleting all objects from memory
183     //
184     // deleting the event deletes all contained vertices, and all particles
185     // contained in those vertices
186     delete evt;
187
188     if( numbad > 0 ) std::cerr << numbad << " errors in testFlow" << std::endl;
189
190     return numbad;
191 }

```

10.9 testHepMC.cc.in

The **HepMC** (p.19) tests can also serve as useful examples based on `example_EventSelection`. Apply an event selection to the events in `testHepMC.input`. Events containing a photon of $p_T > 25$ GeV pass the selection and are written to `"testHepMC.out"`. Add arbitrary PDF information to the good events. Also write events using `IO_AsciiParticles`. Test the new `GenCrossSection` class.

```

1 //-----
2 // testHepMC.cc.in
3 //
4 // garren@fnal.gov, March 2006
5 // based on example_EventSelection
6 // Apply an event selection to the events in testHepMC.input
7 // Events containing a photon of pT > 25 GeV pass the selection
8 // and are written to "testHepMC.out"
9 // Also write events using IO_AsciiParticles
10 //-----
11 //
12
13 #include "HepMC/GenEvent.h"
14 #include "HepMC/GenCrossSection.h"
15 #ifndef HEPMC_IO_ASCII_REMOVED
16 #include "HepMC/IO_Ascii.h"
17 #endif
18 #ifdef HEPMC_HAS_IO_GENEVENT
19 #include "HepMC/IO_GenEvent.h"
20 #endif
21 #include "HepMC/IO_AsciiParticles.h"
22
23 // define methods and classes used by this test
24 #include "IsGoodEvent.h"
25 #include "testHepMCMethods.h"
26
27 void read_testIOGenEvent();
28 void read_variousFormats();
29 void writeWithCrossSection();
30 void readWithCrossSection();
31 void read_nan();
32
33 int main() {
34     read_testIOGenEvent();
35     read_variousFormats();
36     read_nan();
37     writeWithCrossSection();
38     readWithCrossSection();
39     return 0;
40 }
41
42 void read_testIOGenEvent()
43 {
44     std::cout << std::endl;
45     std::cout << "basic IO_GenEvent input and output" << std::endl;
46     // declare an input strategy to read the data produced with the
47     // example_MyPythia - units are GeV and mm
48     HepMC::IO_GenEvent ascii_in("@srcdir/testIOGenEvent.input",std::ios::in);
49     ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
50     // declare another IO_GenEvent for writing out the good events
51     HepMC::IO_GenEvent ascii_out("testHepMC.out",std::ios::out);
52     // declare an output IO_GenEvent for testing precision
53     HepMC::IO_GenEvent prec_out("testHepMCprecision.out",std::ios::out);
54     prec_out.precision(10);
55     // declare an IO_AsciiParticle for output
56     HepMC::IO_AsciiParticles particle_out("testHepMCParticle.out",std::ios::out);

```

```

57 // declare an instance of the event selection predicate
58 IsGoodEvent is_good_event;
59 //.....EVENT LOOP
60 int icount=0;
61 int num_good_events=0;
62 HepMC::GenEvent* evt = ascii_in.read_next_event();
63 while ( evt ) {
64     ++icount;
65     if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
66                             << " its # " << evt->event_number()
67                             << std::endl;
68     if ( is_good_event(evt) ) {
69         particleTypes(evt);
70         ascii_out << evt;
71         particle_out << evt;
72         prec_out << evt;
73         ++num_good_events;
74     }
75
76     // clean up and get next event
77     delete evt;
78     ascii_in >> evt;
79 }
80 //.....PRINT RESULT
81 std::cout << num_good_events << " out of " << icount
82         << " processed events passed the cuts. Finished." << std::endl;
83 }
84
85 void read_variousFormats()
86 {
87     std::cout << std::endl;
88     std::cout << "process varied input" << std::endl;
89     // declare an input strategy
90     HepMC::IO_GenEvent ascii_in("@srcdir/testHepMCVarious.input",std::ios::in);
91     ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
92     // declare another IO_GenEvent for writing out the good events
93     HepMC::IO_GenEvent ascii_out("testHepMCVarious.out",std::ios::out);
94     //.....EVENT LOOP
95     int icount=0;
96     HepMC::GenEvent* evt = ascii_in.read_next_event();
97     while ( evt ) {
98         icount++;
99         double pim;
100         std::cout << "Processing Event Number " << icount
101                 << " its # " << evt->event_number()
102                 << std::endl;
103         ascii_out << evt;
104         // units should be unknown
105         evt->write_units();
106         pim = findPiZero(evt);
107         std::cout << " pizero mass: " << pim << std::endl;
108         // set units to GeV and mm
109         evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
110         evt->write_units();
111         pim = findPiZero(evt);
112         std::cout << " pizero mass: " << pim
113                 << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
114         // convert units to MeV
115         evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
116         evt->write_units();
117         pim = findPiZero(evt);
118         std::cout << " pizero mass: " << pim
119                 << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
120         // clean up and get next event
121         delete evt;
122         ascii_in >> evt;
123     }

```

```

124     //.....PRINT RESULT
125     std::cout << icount << " events processed. Finished." << std::endl;
126 }
127
128 void writeWithCrossSection()
129 {
130     // declare an input strategy to read input data
131     // units are GeV and mm
132     HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
133     ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
134     // declare another IO_GenEvent for writing out some events
135     HepMC::IO_GenEvent ascii_out("testCrossSection.out",std::ios::out);
136     // declare an output stream for printing events
137     std::ofstream xout( "testCrossSection.cout" );
138     // create an empty GenCrossSection object
139     HepMC::GenCrossSection cross;
140     //.....EVENT LOOP
141     int icount=0;
142     const double xs0 = 0.00346;
143     const double xs1 = 0.12;
144     const double xs2 = 33.234;
145     const double xs3 = 459.345;
146     double xserr = 0.0001;
147     HepMC::GenEvent* evt = ascii_in.read_next_event();
148     while ( evt ) {
149         icount++;
150         // use a variety of arbitrary cross section values
151         if( icount < 10 ) {
152             const double xs = xs0 - 1.34 * xserr;
153             cross.set_cross_section( xs, xserr );
154         } else if( icount < 20 ) {
155             const double xs = xs1 - 1.34 * xserr;
156             cross.set_cross_section( xs, xserr );
157         } else if( icount < 30 ) {
158             const double xs = xs2 - 1.34 * xserr;
159             cross.set_cross_section( xs, xserr );
160         } else {
161             const double xs = xs3 - 1.34 * xserr;
162             cross.set_cross_section( xs, xserr );
163         }
164         xserr *= 0.99;
165         if ( icount == 10 ) xserr += 0.01;
166         if ( icount == 20 ) xserr += 0.4;
167         if ( icount == 30 ) xserr += 1.0;
168         // attach this cross section to the event
169         evt->set_cross_section( cross );
170         evt->write_cross_section();
171         if ( icount%20==1 ) {
172             std::cout << "writeWithCrossSection: Processing Event Number " << icount
173                     << " its # " << evt->event_number()
174                     << std::endl;
175             ascii_out << evt;
176             evt->print(xout);
177         }
178         // clean up and get next event
179         delete evt;
180         ascii_in >> evt;
181     }
182     //.....PRINT RESULT
183     std::cout << "writeWithCrossSection processed " << icount << " events. Finished." << std::endl;
184 }
185
186
187 void readWithCrossSection()
188 {
189     // read the file we just wrote
190     HepMC::IO_GenEvent ascii_in("testCrossSection.out",std::ios::in);

```



```

191 // declare another IO_GenEvent for writing out some events
192 HepMC::IO_GenEvent ascii_out("testCrossSection2.out",std::ios::out);
193 //.....EVENT LOOP
194 int icount=0;
195 HepMC::GenEvent* evt = ascii_in.read_next_event();
196 while ( evt ) {
197     ++icount;
198     std::cout << "readWithCrossSection: Processing Event Number " << icount
199                 << " its # " << evt->event_number()
200                 << std::endl;
201     if (evt->cross_section()->cross_section() <= 0) {
202         std::cout << "testReadCrossSection: invalid cross-section!" << std::endl;
203     }
204     ascii_out << evt;
205
206     // clean up and get next event
207     delete evt;
208     ascii_in >> evt;
209 }
210 //.....PRINT RESULT
211 std::cout << "readWithCrossSection processed " << icount << " events. Finished." << std::endl;
212 }
213
214 void read_nan()
215 {
216     // Read an input file that has corrupt information (nan's)
217     //
218     HepMC::IO_GenEvent xin("@srcdir@/testHepMCVarious.input",std::ios::in);
219     HepMC::IO_GenEvent xout("testNaN.out",std::ios::out);
220     //.....EVENT LOOP
221     int icount=0;
222     int invaliddata=0;
223     bool ok = true;
224     std::cout << "----- " << std::endl;
225     std::cout << "Begin NaN test " << std::endl;
226     HepMC::GenEvent* evt = xin.read_next_event();
227     //
228     // To recover from corrupt input, replace "while(evt) {...}"
229     // with "while(ok) { if(evt) {... xin >> evt;} else {...} }"
230     //
231     while ( ok ) {
232         if( evt ) {
233             ++icount;
234             std::cout << "read_nan: Processing Event Number " << icount
235                     << " its # " << evt->event_number()
236                     << std::endl;
237             xout << evt;
238             // clean up and get next event
239             delete evt;
240             xin >> evt;
241         } else if (xin.error_type() == HepMC::IO_Exception::InvalidData ) {
242             ++invaliddata;
243             std::cerr << "INPUT ERROR: " << xin.error_message() << std::endl;
244             // clean up and get next event
245             delete evt;
246             xin >> evt;
247         } else if (invaliddata > 50 ) {
248             std::cerr << "INPUT ERROR: " << xin.error_message() << std::endl;
249             ok = false;
250         } else {
251             ok = false;
252         }
253     }
254     // print status of input stream
255     if ( xin.error_type() != 0 ) {
256         std::cout << "processing of @srcdir@/testHepMCVarious.input ended with error "
257                 << xin.error_type() << std::endl;

```

```
258         std::cout << " --- " << xin.error_message() << std::endl;
259     }
260     std::cout << icount << " events processed and "
261         << invaliddata << " events ignored. Finished."
262         << std::endl;
263     std::cout << "End NaN test " << std::endl;
264     std::cout << "----- " << std::endl;
265 }
266
```

10.10 testHepMCIteration.cc.in

Use Matt's example_EventSelection along with example_UsingIterators to check **HepMC** (p.19) iteration. Apply an event selection to the events in testHepMC.input Events containing a photon of $p_T > 25$ GeV pass the selection. Use iterators on these events.

```

1
2 // testHepMCIteration.cc.in
3 //
4 // garren@fnal.gov, May 2007
5 // Use Matt's example_EventSelection along with example_UsingIterators
6 // to check HepMC iteration.
7 // Apply an event selection to the events in testHepMC.input
8 // Events containing a photon of  $p_T > 25$  GeV pass the selection.
9 // Use iterators on these events.
10
11
12 #include <list>
13
14 #include "HepMC/IO_GenEvent.h"
15 #include "HepMC/IO_AsciiParticles.h"
16 #include "HepMC/GenEvent.h"
17
18 // define methods and classes used by this test
19 #include "IsGoodEvent.h"
20 #include "testHepMCIteration.h"
21
22 bool findW( HepMC::GenEvent* evt, std::ofstream& os);
23 bool simpleIter( HepMC::GenEvent* evt, std::ofstream& os );
24
25 int main() {
26     // declare an input strategy to read the data produced with the
27     // example_MyPythia
28     HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
29     // declare an instance of the event selection predicate
30     IsGoodEvent is_good_event;
31     // define an output stream
32     std::ofstream os( "testHepMCIteration.out" );
33     //.....EVENT LOOP
34     int icount=0;
35     int num_good_events=0;
36     HepMC::GenEvent* evt = ascii_in.read_next_event();
37     HepMC::GenEvent* evcopy;
38     while ( evt ) {
39         icount++;
40         if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
41                                << " its # " << evt->event_number()
42                                << std::endl;
43         // icount of 100 should be the last event
44         if ( icount==100 ) std::cout << "Processing Event Number " << icount
45                                << " its # " << evt->event_number()
46                                << std::endl;
47         evcopy = evt;
48         if ( is_good_event(evcopy) ) {
49             os << "Event " << evcopy->event_number() << " is good " << std::endl;
50             ++num_good_events;
51             // test iterators
52             simpleIter( evcopy, os );
53             findW( evcopy, os );
54         }
55         evcopy->clear();
56
57         // clean up and get next event
58         delete evt;
59         evt = ascii_in.read_next_event();
60     }

```

```

61 //.....PRINT RESULT
62 std::cout << num_good_events << " out of " << icount
63 << " processed events passed the cuts. Finished." << std::endl;
64 }
65
66 bool simpleIter( HepMC::GenEvent* evt, std::ofstream& os )
67 {
68     // use GenEvent::vertex_iterator to fill a list of all
69     // vertices in the event
70     std::list<HepMC::GenVertex*> allvertices;
71     for ( HepMC::GenEvent::vertex_iterator v = evt->vertices_begin();
72           v != evt->vertices_end(); ++v ) {
73         allvertices.push_back(*v);
74     }
75
76     // we could do the same thing with the STL algorithm copy
77     std::list<HepMC::GenVertex*> allvertices2;
78     copy( evt->vertices_begin(), evt->vertices_end(),
79           back_inserter(allvertices2) );
80
81     // fill a list of all final state particles in the event, by requiring
82     // that each particle satisfies the IsFinalState predicate
83     IsFinalState isfinal;
84     std::list<HepMC::GenParticle*> finalstateparticles;
85     for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
86           p != evt->particles_end(); ++p ) {
87         if ( isfinal(*p) ) finalstateparticles.push_back(*p);
88     }
89
90     // an STL-like algorithm called HepMC::copy_if is provided in the
91     // GenEvent.h header to do this sort of operation more easily,
92     // you could get the identical results as above by using:
93     std::list<HepMC::GenParticle*> finalstateparticles2;
94     HepMC::copy_if( evt->particles_begin(), evt->particles_end(),
95                     back_inserter(finalstateparticles2), IsFinalState() );
96
97     // print all photons in the event that satisfy the IsPhoton criteria
98     os << "photons in event " << evt->event_number() << ":" << std::endl;
99     for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
100           p != evt->particles_end(); ++p ) {
101         if ( IsPhoton(*p) ) (*p)->print( os );
102     }
103     return true;
104 }
105
106 bool findW( HepMC::GenEvent* evt, std::ofstream& os )
107 {
108     int num_W=0;
109     // use GenEvent::particle_iterator to find all W's in the event,
110     // then
111     // (1) for each W user the GenVertex::particle_iterator with a range of
112     //     parents to return and print the immediate mothers of these W's.
113     // (2) for each W user the GenVertex::particle_iterator with a range of
114     //     descendants to return and print all descendants of these W's.
115     for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
116           p != evt->particles_end(); ++p ) {
117         if ( IsWBoson(*p) ) {
118             ++num_W;
119             os << "A W boson has been found in event: " << evt->event_number() << std::endl;
120             (*p)->print( os );
121             // return all parents
122             // we do this by pointing to the production vertex of the W
123             // particle and asking for all particle parents of that vertex
124             os << "\t Its parents are: " << std::endl;
125             if ( (*p)->production_vertex() ) {
126                 for ( HepMC::GenVertex::particle_iterator mother
127                       = (*p)->production_vertex()->

```

```

128         particles_begin(HepMC::parents);
129         mother != (*p)->production_vertex()->
130         particles_end(HepMC::parents);
131         ++mother ) {
132             os << "\t";
133             (*mother)->print( os );
134         }
135     }
136
137     // return immediate children
138     os << "\t\t" << "Its children are: " << std::endl;
139     if ( (*p)->end_vertex() ) {
140         for ( HepMC::GenVertex::particle_iterator child =
141             (*p)->end_vertex()->particles_begin(HepMC::children);
142             child != (*p)->end_vertex()->particles_end(HepMC::children);
143             ++child ) {
144             // make a copy
145             HepMC::GenVertex::particle_iterator cp = child;
146             // use the copy and the original
147             os << "\t\t\t (id,barcode,status) "
148                 << (*cp)->pdg_id() << " "
149                 << (*child)->barcode() << " "
150                 << (*cp)->status() << std::endl;
151         }
152     }
153
154     // return all descendants
155     // we do this by pointing to the end vertex of the W
156     // particle and asking for all particle descendants of that vertex
157     os << "\t\t Its descendants are: " << std::endl;
158     if ( (*p)->end_vertex() ) {
159         for ( HepMC::GenVertex::particle_iterator des
160             = (*p)->end_vertex()->
161             particles_begin(HepMC::descendants);
162             des != (*p)->end_vertex()->
163             particles_end(HepMC::descendants);
164             ++des ) {
165             os << "\t\t";
166             (*des)->print( os );
167         }
168     }
169     } // if IsWBoson
170 } // end particle loop
171 return true;
172 }
173

```

10.11 testHerwigCopies.cc

Multiple events in memory at the same time

```

1
2 // testHerwigCopies.cc
3 //
4 // garren@fnal.gov, January 2008
5 // Multiple events in memory at the same time
6
7
8 #include <fstream>
9 #include <iostream>
10 #include "HepMC/HerwigWrapper.h"
11 #include "HepMC/IO_HERWIG.h"
12 #include "HepMC/GenEvent.h"
13 #include "HepMC/CompareGenEvent.h"
14 #include "HepMC/HEPEVT_Wrapper.h"
15 #include "HerwigHelper.h"
16
17 int main() {
18     //
19     //.....HEPEVT
20     // Herwig 6.4 uses HEPEVT with 4000 entries and 8-byte floating point
21     // numbers. We need to explicitly pass this information to the
22     // HEPEVT_Wrapper.
23     //
24     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
25     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
26     //
27     //.....INITIALIZATIONS
28
29     hwproc.PBEAM1 = 7000.; // energy of beam1
30     hwproc.PBEAM2 = 7000.; // energy of beam2
31     // 1610 = gg->H-> WW, 1706 = qq->ttbar, 2510 = ttH -> ttWW
32     hwproc.IPROC = 1706; // qq -> ttbar production
33     hwproc.MAXEV = 50; // number of events
34     // tell it what the beam particles are:
35     for ( unsigned int i = 0; i < 8; ++i ) {
36         hwbmch.PART1[i] = (i < 1) ? 'P' : ' ';
37         hwbmch.PART2[i] = (i < 1) ? 'P' : ' ';
38     }
39     hwigin(); // INITIALISE OTHER COMMON BLOCKS
40     hwevnt.MAXPR = 0; // number of events to print
41     hwuinc(); // compute parameter-dependent constants
42     hweini(); // initialise elementary process
43
44     //.....HepMC INITIALIZATIONS
45     //
46     // Instantiate an IO strategy for reading from HEPEVT.
47     HepMC::IO_HERWIG hepevtio;
48     //
49     // open some output files
50     std::ofstream out1( "testHerwigOriginals.dat" );
51     std::ofstream out2( "testHerwigCopies1.dat" );
52     std::ofstream out3( "testHerwigCopies2.dat" );
53     //
54     //.....EVENT LOOP
55     for ( int i = 1; i <= hwproc.MAXEV; i++ ) {
56         if ( i%50==1 ) std::cout << "Processing Event Number "
57                                 << i << std::endl;
58         // initialise event
59         hwuine();
60         // generate hard subprocess
61         hwepro();
62         // generate parton cascades
63         hwbgen();

```

```

64         // do heavy object decays
65         hwdhob();
66         // do cluster formation
67         hwcfor();
68         // do cluster decays
69         hwcdec();
70         // do unstable particle decays
71         hwdhad();
72         // do heavy flavour hadron decays
73         hwdhvy();
74         // add soft underlying event if needed
75         hwmevt();
76         // finish event
77         hwufne();
78         HepMC::GenEvent* evt = hepevtio.read_next_event();
79         // herwig uses GeV and mm
80         evt->use_units( HepMC::Units::GEV, HepMC::Units::MM);
81         // set cross section information
82         evt->set_cross_section( getHerwigCrossSection(i) );
83         // add some information to the event
84         evt->set_event_number(i);
85         evt->set_signal_process_id(20);
86         //
87         //.....make some copies
88         evt->print(out1);
89         HepMC::GenEvent ec = (*evt);
90         ec.print(out2);
91         HepMC::GenEvent* evt4 = new HepMC::GenEvent(*evt);
92         evt4->print(out3);
93         if( !compareGenEvent(evt,evt4) ) {
94             std::cerr << "testHerwigCopies: GenEvent comparison fails at event "
95                       << evt->event_number() << std::endl;
96             return -1;
97         }
98
99         // we also need to delete the created event from memory
100        delete evt;
101        delete evt4;
102    }
103    //.....TERMINATION
104    hwefin();
105    std::cout << "testHerwigCopies: event comparison is successful" << std::endl;
106
107    return 0;
108 }

```

10.12 testMass.cc.in

Read events from testIOGenEvent.input Select events containing a photon of $p_T > 25$ GeV Add arbitrary PDF information to one of the good events Write the selected events and read them back in using an istream

```

1 //-----
2 // testMass.cc.in
3 //
4 // garren@fnal.gov, March 2006
5 // Read events written by example_MyPythia.cc
6 // Select events containing a photon of  $p_T > 25$  GeV
7 // Add arbitrary PDF information to one of the good events
8 // Add arbitrary HeavyIon information to one of the good events
9 // Write the selected events and read them back in using an istream
10 //-----
11
12 #include <cmath>          // for min()
13
14 #include "HepMC/IO_GenEvent.h"
15 #include "HepMC/GenEvent.h"
16 #include "HepMC/Version.h"
17
18 // define methods and classes used by this test
19 #include "IsGoodEvent.h"
20
21 void massInfo( const HepMC::GenEvent* );
22
23 int main() {
24     // read and process the input file
25     {
26         // declare an input strategy to read the data produced with the
27         // example_MyPythia
28         HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
29         ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
30         // declare another IO_GenEvent for output
31         HepMC::IO_GenEvent ascii_out("testMass1.out",std::ios::out);
32         // declare an instance of the event selection predicate
33         IsGoodEvent is_good_event;
34         // send version to standard output
35         HepMC::version();
36         //.....EVENT LOOP
37         int icount=0;
38         int num_good_events=0;
39         double x1=0., x2=0., q=0., xf1=0., xf2=0.;
40         HepMC::GenEvent* evt = ascii_in.read_next_event();
41         while ( evt ) {
42             icount++;
43             if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
44                                     << " its # " << evt->event_number()
45                                     << std::endl;
46             if ( is_good_event(evt) ) {
47                 if (num_good_events == 0 ) {
48                     // add some arbitrary PDF information
49                     x1 = std::min(0.8, 0.07 * icount);
50                     x2 = 1-x1;
51                     q = 1.69 * icount;
52                     // use beam momentum
53                     if( evt->valid_beam_particles() ) {
54                         HepMC::GenParticle* bpl = evt->beam_particles().first;
55                         xf1 = x1*bpl->momentum().rho();
56                         xf2 = x2*bpl->momentum().rho();
57                     } else {
58                         xf1 = x1*0.34;
59                         xf2 = x2*0.34;

```



```

60         }
61         // provide optional pdf set id numbers
62         // (two ints at the end of the constructor)
63         HepMC::PdfInfo pdf( 2, 3, x1, x2, q, xf1, xf2, 230, 230);
64         evt->set_pdf_info(pdf);
65         // add some arbitrary HeavyIon information
66         HepMC::HeavyIon ion(23,11,12,15,3,5,0,0,0,0.0145);
67         evt->set_heavy_ion( ion );
68     }
69     ascii_out << evt;
70     ++num_good_events;
71 }
72
73     // clean up and get next event
74     delete evt;
75     ascii_in >> evt;
76 }
77 //.....PRINT RESULT
78 std::cout << num_good_events << " out of " << icount
79     << " processed events passed the cuts. Finished." << std::endl;
80 }
81 // now read the file we just created
82 {
83     // declare an input strategy
84     const char infile[] = "testMass1.out";
85     std::ifstream istr( infile );
86     if( !istr ) {
87         std::cerr << "testMass: cannot open " << infile << std::endl;
88         exit(-1);
89     }
90     HepMC::IO_GenEvent xin(istr);
91     // declare another IO_GenEvent for output
92     HepMC::IO_GenEvent xout("testMass2.out",std::ios::out);
93     //.....EVENT LOOP
94     int ixin=0;
95     HepMC::GenEvent* evt = xin.read_next_event();
96     while ( evt ) {
97         ixin++;
98         std::cout << "reading Event " << ixin << std::endl;
99         xout << evt;
100         // look at mass info
101         massInfo(evt);
102
103         // clean up and get next event
104         delete evt;
105         xin >> evt;
106     }
107     //.....PRINT RESULT
108     std::cout << ixin
109         << " events in the second pass. Finished." << std::endl;
110 }
111 }
112
113 void massInfo( const HepMC::GenEvent* e )
114 {
115     double gm, m, d;
116     for ( HepMC::GenEvent::particle_const_iterator p = e->particles_begin(); p != e->particles_end();
117         ++p ) {
118
119         gm = (*p)->generated_mass();
120         m = (*p)->momentum().m();
121         d = fabs(m-gm);
122         if( d > 1.0e-5 ) {
123             std::cout << "Event " << e->event_number()
124                 << " Particle " << (*p)->barcode()
125                 << " " << (*p)->pdg_id()
126                 << " generated mass " << gm

```

```
127         << " mass from momentum " << m
128         << " difference " << d << std::endl;
129     }
130 }
131 }
```

10.13 testMultipleCopies.cc.in

Multiple events in memory at the same time run with valgrind or some other leak checker

```

1
2 // testMultipleCopies.cc.in
3 //
4 // garren@fnal.gov, January 2008
5 // Multiple events in memory at the same time
6 // run with valgrind or some other leak checker
7 //
8 //
9
10 #include <fstream>
11
12 #include "HepMC/IO_GenEvent.h"
13 #include "HepMC/GenEvent.h"
14 #include "HepMC/CompareGenEvent.h"
15
16 // define methods and classes used by this test
17 #include "IsGoodEvent.h"
18
19 int main() {
20     // use output file
21     std::ofstream os( "testMultipleCopies.out" );
22     {
23         // declare an input strategy
24         HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
25         // declare another input strategy
26         HepMC::IO_GenEvent ascii_in2("@srcdir@/testHepMCVarious.input",std::ios::in);
27         std::ofstream out1( "testMultipleOriginals.out" );
28         std::ofstream out2( "testMultipleCopies1.out" );
29         std::ofstream out3( "testMultipleCopies2.out" );
30         // declare an instance of the event selection predicate
31         IsGoodEvent is_good_event;
32
33         //.....EVENT LOOP
34         int icount=0;
35         int num_good_events=0;
36         int icnt;
37         HepMC::GenEvent* evt1 = ascii_in.read_next_event();
38         HepMC::GenEvent* evt2 = ascii_in2.read_next_event();
39         HepMC::GenEvent* evt3 = ascii_in.read_next_event();
40
41         while ( evt1 && evt2 ) {
42             icount++;
43             if ( icount%50==1 ) os << "Processing Event Number " << icount
44                                 << " stream 1 # " << evt1->event_number()
45                                 << " stream 2 # " << evt2->event_number()
46                                 << std::endl;
47
48             if ( is_good_event(evt1) ) {
49
50                 os << "good event in stream 1 # "
51                     << evt1->event_number() << std::endl;
52                 evt1->print(out1);
53                 ++num_good_events;
54                 HepMC::GenEvent ec = (*evt1);
55                 ec.print(out3);
56                 icnt=0;
57                 for ( HepMC::GenEvent::particle_const_iterator p1 = ec.particles_begin();
58                     p1 != ec.particles_end(); ++p1 ) {
59                     ++icnt;
60                     os << "particle " << icnt << " barcode " <<(*p1)->barcode() << std::endl;
61                 }
62                 HepMC::GenEvent* evt4 = new HepMC::GenEvent(*evt1);

```

```

63         evt4->print(out2);
64         if( !compareGenEvent(evt1,evt4) ) { return -1; }
65         delete evt4;
66     }
67
68     // clean up and get next events
69     delete evt1;
70     delete evt2;
71     ascii_in >> evt1;
72     ascii_in2 >> evt2;
73 }
74 // might have either evt1 or evt2 still in memory, cleanup here
75 delete evt1;
76 delete evt2;
77 delete evt3;
78
79 //.....PRINT RESULT
80 os << std::endl;
81 os << num_good_events << " out of " << icount
82   << " processed events passed the cuts." << std::endl;
83 os << std::endl;
84 os << " GenEvent copy constructor passes the test" << std::endl;
85 os << std::endl;
86 }
87
88 // test operator= and swap
89 {
90     // declare an input strategy
91     HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
92     //
93     HepMC::GenEvent* evt5 = ascii_in.read_next_event();
94     HepMC::GenEvent* evt6 = new HepMC::GenEvent();
95     os << "event number for evt5: " << evt5->event_number() << std::endl;
96     os << "event number for evt6: " << evt6->event_number() << std::endl;
97     // copy GenEvent object
98     (*evt6) = (*evt5);
99     if( !compareGenEvent(evt5,evt6) ) { return -4; }
100    delete evt5;
101    os << "event number for evt6 after copy: " << evt6->event_number() << std::endl;
102    os << std::endl;
103    delete evt6;
104    os << " GenEvent operator= passes the test" << std::endl;
105    os << std::endl;
106
107    evt5 = ascii_in.read_next_event();
108    evt6 = ascii_in.read_next_event();
109    HepMC::GenEvent* evt7 = new HepMC::GenEvent(*evt5);
110    HepMC::GenEvent* evt8 = new HepMC::GenEvent(*evt6);
111    os << "event number for evt5: " << evt5->event_number() << std::endl;
112    os << "event number for evt6: " << evt6->event_number() << std::endl;
113    os << "before swap, evt5 has: " << evt5->vertices_size() << " vertices and "
114      << evt5->particles_size() << " particles" << std::endl;
115    os << "before swap, evt6 has: " << evt6->vertices_size() << " vertices and "
116      << evt6->particles_size() << " particles" << std::endl;
117    os << "before swap, evt7 has: " << evt7->vertices_size() << " vertices and "
118      << evt7->particles_size() << " particles" << std::endl;
119    os << "before swap, evt8 has: " << evt8->vertices_size() << " vertices and "
120      << evt8->particles_size() << " particles" << std::endl;
121    (*evt6).swap(*evt5);
122    os << "event number for evt5 after swap: " << evt5->event_number() << std::endl;
123    os << "event number for evt6 after swap: " << evt6->event_number() << std::endl;
124    // evt6 should now match evt7
125    os << "after swap, evt6 has: " << evt6->vertices_size() << " vertices and "
126      << evt6->particles_size() << " particles" << std::endl;
127    os << "after swap, evt7 has: " << evt7->vertices_size() << " vertices and "
128      << evt7->particles_size() << " particles" << std::endl;
129    if( !compareGenEvent(evt6,evt7) ) { return -6; }

```

```
130         // evt5 should now match evt8
131         os << "after swap, evt5 has: " << evt5->vertices_size() << " vertices and "
132           << evt5->particles_size() << " particles" << std::endl;
133         os << "after swap, evt8 has: " << evt8->vertices_size() << " vertices and "
134           << evt8->particles_size() << " particles" << std::endl;
135         if( !compareGenEvent(evt5,evt8) ) { return -5; }
136         // cleanup
137         delete evt5;
138         delete evt6;
139         delete evt7;
140         delete evt8;
141         os << std::endl;
142         os << " GenEvent swap passes the test" << std::endl;
143         os << std::endl;
144     }
145     return 0;
146 }
```

10.14 testPrintBug.cc

Thanks to Bob McElrath and Frank Siegert for this test

```
1 //
2 // Thanks to Bob McElrath and Frank Siegert for this test
3 //
4
5 #include <fstream>
6
7 #include "HepMC/GenEvent.h"
8 #include "HepMC/SimpleVector.h"
9
10 int main(int argc, char* argv[])
11 {
12     HepMC::GenEvent* p_event;
13
14     p_event = new HepMC::GenEvent();
15
16     // define an output stream
17     std::ofstream os( "testPrintBug.out" );
18
19     for(int i=0; i<10; i++) {
20         HepMC::FourVector vector(1.0,1.0,1.0,1.0);
21         HepMC::GenVertex* vertex = new HepMC::GenVertex(vector,i);
22         for(int j=0; j<3; j++) {
23             HepMC::GenParticle* particle = new HepMC::GenParticle(vector,1,2);
24             vertex->add_particle_in(particle);
25         }
26         for(int j=0; j<3; j++) {
27             HepMC::GenParticle* particle = new HepMC::GenParticle(vector,1,2);
28             vertex->add_particle_out(particle);
29         }
30         p_event->add_vertex(vertex);
31     }
32     p_event->print(os);
33     // cleanup
34     delete p_event;
35     return 0;
36 }
```

10.15 testPythiaCopies.cc

Multiple events in memory at the same time

```

1
2 // testPythiaCopies.cc
3 //
4 // garren@fnal.gov, January 2008
5 // Multiple events in memory at the same time
6
7
8 #include <fstream>
9 #include <iostream>
10 #include "HepMC/PythiaWrapper.h"
11 #include "HepMC/IO_HEPEVT.h"
12 #include "HepMC/GenEvent.h"
13 #include "HepMC/CompareGenEvent.h"
14 #include "PythiaHelper.h"
15
16 int main() {
17     //
18     //.....HEPEVT
19     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
20     // numbers. We need to explicitly pass this information to the
21     // HEPEVT_Wrapper.
22     //
23     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
24     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
25     //
26     //.....PYTHIA INITIALIZATIONS
27     initPythia();
28     //
29     //.....HepMC INITIALIZATIONS
30     //
31     // Instantiate an IO strategy for reading from HEPEVT.
32     HepMC::IO_HEPEVT hepevtio;
33     //
34     // open some output files
35     std::ofstream out1( "testPythiaOriginals.dat" );
36     std::ofstream out2( "testPythiaCopies1.dat" );
37     std::ofstream out3( "testPythiaCopies2.dat" );
38     //
39     //.....EVENT LOOP
40     for ( int i = 1; i <= 50; i++ ) {
41         if ( i%50==1 ) std::cout << "Processing Event Number "
42             << i << std::endl;
43         call_pyevnt(); // generate one event with Pythia
44         // pythia pyhepc routine convert common PYJETS in common HEPEVT
45         call_pyhepc( 1 );
46         HepMC::GenEvent* evt = hepevtio.read_next_event();
47         // pythia uses GeV and mm
48         evt->use_units( HepMC::Units::GEV, HepMC::Units::MM);
49         // set number of multi parton interactions
50         evt->set_mpi( pypars.msti[31-1] );
51         // set cross section information
52         evt->set_cross_section( getPythiaCrossSection() );
53         //
54         //.....make some copies
55         evt->print(out1);
56         HepMC::GenEvent ec = (*evt);
57         ec.print(out2);
58         HepMC::GenEvent* evt4 = new HepMC::GenEvent(*evt);
59         evt4->print(out3);
60         if( !compareGenEvent(evt,evt4) ) {
61             std::cerr << "testPythiaCopies: GenEvent comparison fails at event "
62                 << evt->event_number() << std::endl;
63             return -1;

```

```
64         }
65         //
66         // now delete the created events from memory
67         delete evt;
68         delete evt4;
69     }
70     //.....TERMINATION
71     // write out some information from Pythia to the screen
72     call_pystat( 1 );
73     std::cout << "testPythiaCopies: event comparison is successful" << std::endl;
74
75     return 0;
76 }
77
78
79
```


10.16 testSimpleVector.cc

Exercise all the vector methods

```

1 //
2 // First pass - simply exercise all the vector methods
3 //
4 #include <iostream>
5
6 #include "HepMC/SimpleVector.h"
7
8 int main()
9 {
10     // ThreeVector
11     HepMC::ThreeVector vector3;
12     HepMC::ThreeVector v3(1.1,2.2,3.3);
13     HepMC::ThreeVector vx(1.34);
14
15     HepMC::ThreeVector v3copy( v3 );
16
17     double eps = 1.e-15; // allowed differnce between doubles
18     int numbad = 0;
19
20     double x = v3.x();
21     double y = v3.y();
22     double z = v3.z();
23     double p2 = v3.perp2();
24     double pt = v3.perp();
25     double r = v3.r();
26     double th = v3.theta();
27     double ph = v3.phi();
28     double mag = std::sqrt(x*x + y*y + z*z);
29     double pperp = std::sqrt(x*x + y*y);
30
31     vx.set(1., 2., 3.);
32     vx.setX(1.1);
33     vx.setY(2.3);
34     vx.setZ(4.4);
35     vx.setPhi(0.12);
36     vx.setTheta(0.54);
37
38     vector3 = v3;
39
40     if( fabs( mag - r ) > eps ) {
41         std::cout << "different ThreeVector magnitude: " << mag << " " << r << std::endl;
42         std::cout << "difference is : " << ( mag - r ) << std::endl;
43         ++numbad;
44     }
45     if( fabs( pperp - pt ) > eps ) {
46         std::cout << "different ThreeVector Pt: " << pperp << " " << pt << std::endl;
47         std::cout << "difference is : " << ( pperp - pt ) << std::endl;
48         ++numbad;
49     }
50
51     if( v3 == vector3 ) {
52     } else {
53         ++numbad;
54         std::cout << "vectors v3 and vector3 are different" << std::endl;
55     }
56     if( v3 != v3copy ) {
57         ++numbad;
58         std::cout << "vectors v3 and v3copy are different" << std::endl;
59     }
60
61     // FourVector
62     HepMC::FourVector vector;

```

```

63  HepMC::FourVector v4(1.1,2.2,3.3,4.4);
64  HepMC::FourVector vt(1.34);
65
66  HepMC::FourVector vectorcopy( v4 );
67  vector = v4;
68
69  double px = v4.px();
70  double py = v4.py();
71  double pz = v4.pz();
72  double e  = v4.e();
73  x = vectorcopy.x();
74  y = vectorcopy.y();
75  z = vectorcopy.z();
76  double t = vectorcopy.t();
77
78  p2 = v4.perp2();
79  pt = v4.perp();
80  th = v4.theta();
81  ph = v4.phi();
82  r = v4.rho();
83  double masssq1 = v4.m2();
84  double mass1 = v4.m();
85  double pr1 = v4.pseudoRapidity();
86  double eta1 = v4.eta();
87  double masssq2 = vector.m2();
88  double mass2 = vector.m();
89  double pr2 = vector.pseudoRapidity();
90  double eta2 = vector.eta();
91
92  vt.set(1., 2., 3., 5.5);
93  vt.setX(1.1);
94  vt.setY(2.3);
95  vt.setZ(4.4);
96  vt.setT(6.5);
97  vt.setPx(3.1);
98  vt.setPy(2.2);
99  vt.setPz(-1.1);
100 vt.setE(5.4);
101
102 mag = std::sqrt(x*x + y*y + z*z);
103 pperp = std::sqrt(x*x + y*y);
104 if( fabs( mag - r ) > eps ) {
105     std::cout << "different FourVector magnitude: " << mag << " " << r << std::endl;
106     std::cout << "difference is : " << ( mag - r ) << std::endl;
107     ++numbad;
108 }
109 if( fabs( pperp - pt ) > eps ) {
110     std::cout << "different FourVector Pt: " << pperp << " " << pt << std::endl;
111     std::cout << "difference is : " << ( pperp - pt ) << std::endl;
112     ++numbad;
113 }
114
115 if( px != x ) {
116     std::cout << "different X values: " << px << " " << x << std::endl;
117     ++numbad;
118 }
119 if( py != y ) {
120     std::cout << "different Y values: " << py << " " << y << std::endl;
121     ++numbad;
122 }
123 if( pz != z ) {
124     std::cout << "different Z values: " << pz << " " << z << std::endl;
125     ++numbad;
126 }
127 if( e != t ) {
128     std::cout << "different E values: " << e << " " << t << std::endl;
129     ++numbad;

```

```
130     }
131     if( fabs( masssq1 - masssq2 ) > eps ) {
132         std::cout << "different mass sq values: " << masssq1 << " " << masssq2 << std::endl;
133         std::cout << "difference is : " << ( masssq1 - masssq2 ) << std::endl;
134         ++numbad;
135     }
136     if( fabs( mass1 - mass2 ) > eps ) {
137         std::cout << "different mass values: " << mass1 << " " << mass2 << std::endl;
138         std::cout << "difference is : " << ( mass1 - mass2 ) << std::endl;
139         ++numbad;
140     }
141     if( fabs( pr1 - pr2 ) > eps ) {
142         std::cout << "different pseudorapidity values: " << pr1 << " " << pr2 << std::endl;
143         std::cout << "difference is : " << ( pr1 - pr2 ) << std::endl;
144         ++numbad;
145     }
146     if( fabs( eta1 - eta2 ) > eps ) {
147         std::cout << "different eta values: " << eta1 << " " << eta2 << std::endl;
148         std::cout << "difference is : " << ( eta1 - eta2 ) << std::endl;
149         ++numbad;
150     }
151     if( v4 == vector ) {
152     } else {
153         std::cout << "vectors v and vector are different" << std::endl;
154         ++numbad;
155     }
156     if( v4 != vectorcopy ) {
157         std::cout << "vectors v and vectorcopy are different" << std::endl;
158         ++numbad;
159     }
160
161     return numbad;
162 }
```

10.17 testStreamIO.cc.in

Use streaming IO to read and write a file

```

1
2 // testStreamIO.cc.in
3 //
4 // garren@fnal.gov, March 2006
5 //
6 // The same as testHepMC, but using the IO stream directly
7 //
8 //
9
10 #include <fstream>
11
12 #include "HepMC/GenEvent.h"
13 #include "HepMC/IO_AsciiParticles.h"
14 #ifdef HEPMC_HAS_IO_GENEVENT
15 #include "HepMC/IO_GenEvent.h"
16 #endif
17 #include "HepMC/Version.h"
18 #include "HepMC/IO_Exception.h"
19
20 // define methods and classes used by this test
21 #include "IsGoodEvent.h"
22 #include "testHepMCMethods.h"
23
24 void read_testIOGenEvent();
25 void read_variousFormats();
26 void write_to_stream();
27 void write_to_stream3();
28 void read_from_stream4();
29
30 int main() {
31     write_to_stream();
32     read_testIOGenEvent();
33     read_variousFormats();
34     write_to_stream3();
35     read_from_stream4();
36     return 0;
37 }
38
39 void write_to_stream()
40 {
41     std::cout << std::endl;
42     std::cout << "basic IO_GenEvent input with streaming output" << std::endl;
43     // declare an input strategy to read the data produced with the
44     // example_MyPythia - units are GeV and mm
45     HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input", std::ios::in);
46     ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
47     // declare an output stream
48     const char outfile[] = "testStreamIO.out";
49     std::ofstream ascii_out( outfile );
50     if( !ascii_out ) {
51         std::cerr << "cannot open " << outfile << std::endl;
52         exit(-1);
53     }
54     ascii_out.precision(16);
55     HepMC::write_HepMC_IO_block_begin( ascii_out );
56     // declare an instance of the event selection predicate
57     IsGoodEvent is_good_event;
58     //.....EVENT LOOP
59     int icount=0;
60     int num_good_events=0;
61     HepMC::GenEvent* evt = ascii_in.read_next_event();
62     while ( evt ) {
63         icount++;

```

```

64         if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
65                                     << " its # " << evt->event_number()
66                                     << std::endl;
67         if ( is_good_event(evt) ) {
68             ++num_good_events;
69             particleTypes( evt );
70             ascii_out << (*evt);
71         }
72
73         // clean up and get next event
74         delete evt;
75         ascii_in >> evt;
76     }
77     HepMC::write_HepMC_IO_block_end( ascii_out );
78     //.....PRINT RESULT
79     std::cout << num_good_events << " out of " << icount
80             << " processed events passed the cuts. Finished." << std::endl;
81 }
82
83 void read_testIOGenEvent()
84 {
85     std::cout << std::endl;
86     std::cout << "streaming input and output" << std::endl;
87     // input units are GeV and mm
88     const char infile[] = "@srcdir@/testIOGenEvent.input";
89     std::ifstream is( infile );
90     if( !is ) {
91         std::cerr << "cannot open " << infile << std::endl;
92         exit(-1);
93     }
94     // declare an output stream
95     const char outfile[] = "testStreamIO2.out";
96     std::ofstream ascii_out( outfile );
97     if( !ascii_out ) {
98         std::cerr << "cannot open " << outfile << std::endl;
99         exit(-1);
100    }
101    ascii_out.precision(16);
102    HepMC::write_HepMC_IO_block_begin( ascii_out );
103    // declare another output stream to test precision
104    const char poutfile[] = "testStreamIOprecision.out";
105    std::ofstream pout( poutfile );
106    if( !pout ) {
107        std::cerr << "cannot open " << poutfile << std::endl;
108        exit(-1);
109    }
110    pout.precision(10);
111    // declare an IO_AsciiParticle for output
112    HepMC::IO_AsciiParticles particle_out("testStreamIOParticle.out",std::ios::out);
113    // declare an instance of the event selection predicate
114    IsGoodEvent is_good_event;
115    //.....EVENT LOOP
116    int icount=0;
117    int num_good_events=0;
118    HepMC::GenEvent evt;
119    while ( is ) {
120        // WARNING - we are not using pointers, so this could be an empty event
121        is >> evt;
122        // make sure this is a valid event
123        if( evt.is_valid() ) {
124            ++icount;
125            if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
126                                << " its # " << evt.event_number()
127                                << std::endl;
128            if ( is_good_event( &evt ) ) {
129                ++num_good_events;
130                particleTypes(&evt);

```

```

131         ascii_out << evt;
132         pout << evt;
133         // We must explicitly create the pointer if we want to use this event
134         // with any IO strategy (e.g., IO_AsciiParticles)
135         HepMC::GenEvent* pevt= &evt;
136         particle_out << pevt;
137     }
138 }
139 }
140 HepMC::write_HepMC_IO_block_end( ascii_out );
141 //.....PRINT RESULT
142 std::cout << num_good_events << " out of " << icount
143         << " processed events passed the cuts. Finished." << std::endl;
144 }
145
146 void read_variousFormats()
147 {
148     std::cout << std::endl;
149     std::cout << "process varied input" << std::endl;
150     // declare an input stream
151     const char infile[] = "@srcdir@/testHepMCVarious.input";
152     std::ifstream is( infile );
153     if( !is ) {
154         std::cerr << "cannot open " << infile << std::endl;
155         exit(-1);
156     }
157     // set input units
158     HepMC::set_input_units( is, HepMC::Units::GEV, HepMC::Units::MM );
159     // declare an output stream
160     const char outfile[] = "testStreamIOVarious.out";
161     std::ofstream ascii_out( outfile );
162     if( !ascii_out ) {
163         std::cerr << "cannot open " << outfile << std::endl;
164         exit(-1);
165     }
166     ascii_out.precision(16);
167     HepMC::write_HepMC_IO_block_begin( ascii_out );
168     //.....EVENT LOOP
169     int icount=0, ibad=0;
170     HepMC::GenEvent evt;
171     while ( is ) {
172         // we have to do our own try/catch blocks
173         try {
174             is >> evt;
175         }
176         catch (HepMC::IO_Exception& e) {
177             evt.clear();
178             ++ibad;
179         }
180         // WARNING - we are not using pointers, so this could be an empty event
181         // make sure this is a valid event
182         if( evt.is_valid() ) {
183             icount++;
184             double pim;
185             std::cout << "Processing Event Number " << icount
186                     << " its # " << evt.event_number()
187                     << std::endl;
188             ascii_out << evt;
189             // units should be unknown
190             evt.write_units();
191             pim = findPiZero(&evt);
192             std::cout << " pizero mass: " << pim << std::endl;
193             // set units to GeV and mm
194             evt.use_units(HepMC::Units::GEV, HepMC::Units::MM);
195             evt.write_units();
196             pim = findPiZero(&evt);
197             std::cout << " pizero mass: " << pim

```

```

198         << " " << HepMC::Units::name( evt.momentum_unit() ) << std::endl;
199         // convert units to MeV
200         evt.use_units(HepMC::Units::MEV, HepMC::Units::MM);
201         evt.write_units();
202         pim = findPiZero(&evt);
203         std::cout << " pizero mass: " << pim
204         << " " << HepMC::Units::name( evt.momentum_unit() ) << std::endl;
205     }
206 }
207 HepMC::write_HepMC_IO_block_end( ascii_out );
208 //.....PRINT RESULT
209 std::cout << icount << " valid events processed. " ;
210 std::cout << ibad << " invalid events processed. Finished." << std::endl;
211 }
212
213 void write_to_stream3()
214 {
215     std::cout << std::endl;
216     std::cout << "basic IO_GenEvent input with streaming output using member function" << std::endl;
217     // declare an input strategy to read the data produced with the
218     // example_MyPythia - units are GeV and mm
219     HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
220     ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
221     // declare an output stream
222     const char outfile[] = "testStreamIO3.out";
223     std::ofstream ascii_out( outfile );
224     if( !ascii_out ) {
225         std::cerr << "cannot open " << outfile << std::endl;
226         exit(-1);
227     }
228     ascii_out.precision(16);
229     HepMC::write_HepMC_IO_block_begin( ascii_out );
230     // declare an instance of the event selection predicate
231     IsGoodEvent is_good_event;
232     //.....EVENT LOOP
233     int icount=0;
234     int num_good_events=0;
235     HepMC::GenEvent* evt = ascii_in.read_next_event();
236     while ( evt ) {
237         icount++;
238         if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
239         << " its # " << evt->event_number()
240         << std::endl;
241         if ( is_good_event(evt) ) {
242             ++num_good_events;
243             particleTypes( evt );
244             evt->write(ascii_out);
245         }
246
247         // clean up and get next event
248         delete evt;
249         ascii_in >> evt;
250     }
251     HepMC::write_HepMC_IO_block_end( ascii_out );
252     //.....PRINT RESULT
253     std::cout << num_good_events << " out of " << icount
254     << " processed events passed the cuts. Finished." << std::endl;
255 }
256
257 void read_from_stream4()
258 {
259     std::cout << std::endl;
260     std::cout << "streaming input and output using member functions" << std::endl;
261     // input units are GeV and mm
262     const char infile[] = "@srcdir@/testIOGenEvent.input";
263     std::ifstream is( infile );
264     if( !is ) {

```

```
265     std::cerr << "cannot open " << infile << std::endl;
266     exit(-1);
267 }
268 // declare an output stream
269 const char outfile[] = "testStreamIO4.out";
270 std::ofstream ascii_out( outfile );
271 if( !ascii_out ) {
272     std::cerr << "cannot open " << outfile << std::endl;
273     exit(-1);
274 }
275 ascii_out.precision(16);
276 HepMC::write_HepMC_IO_block_begin( ascii_out );
277 // declare an instance of the event selection predicate
278 IsGoodEvent is_good_event;
279 //.....EVENT LOOP
280 int icount=0;
281 int num_good_events=0;
282 HepMC::GenEvent evt;
283 while ( is ) {
284     // WARNING - we are not using pointers, so this could be an empty event
285     evt.read(is);
286     // make sure this is a valid event
287     if( evt.is_valid() ) {
288         ++icount;
289         if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
290                                     << " its # " << evt.event_number()
291                                     << std::endl;
292         if ( is_good_event( &evt ) ) {
293             ++num_good_events;
294             particleTypes(&evt);
295             evt.write(ascii_out);
296         }
297     }
298 }
299 HepMC::write_HepMC_IO_block_end( ascii_out );
300 //.....PRINT RESULT
301 std::cout << num_good_events << " out of " << icount
302           << " processed events passed the cuts. Finished." << std::endl;
303 }
```


10.18 testUnits.cc

Test MomentumUnits and PositionUnits Make sure set and change methods work as expected.

```

1 //
2 // Test Units
3 //
4 #include <iostream>
5
6 #include "HepMC/Units.h"
7
8 int main()
9 {
10
11     int err = 0;
12     double cf;
13
14     std::cout << "Default units: " << HepMC::Units::name(HepMC::Units::default_momentum_unit())
15         << " " << HepMC::Units::name(HepMC::Units::default_length_unit()) << std::endl;
16
17     // check momentum conversion factors
18     cf = conversion_factor( HepMC::Units::GEV, HepMC::Units::GEV );
19     if( cf != 1 ) {
20         ++err;
21         std::cerr << "wrong conversion factor " << cf
22             << " for GEV to GEV - should be 1 \n";
23     }
24     cf = conversion_factor( HepMC::Units::MEV, HepMC::Units::MEV );
25     if( cf != 1 ) {
26         ++err;
27         std::cerr << "wrong conversion factor " << cf
28             << " for MEV to MEV - should be 1 \n";
29     }
30     cf = conversion_factor( HepMC::Units::MEV, HepMC::Units::GEV );
31     if( cf != 0.001 ) {
32         ++err;
33         std::cerr << "wrong conversion factor " << cf
34             << " for MEV to GEV - should be 0.001 \n";
35     }
36     cf = conversion_factor( HepMC::Units::GEV, HepMC::Units::MEV );
37     if( cf != 1000.0 ) {
38         ++err;
39         std::cerr << "wrong conversion factor " << cf
40             << " for GEV to MEV - should be 1000 \n";
41     }
42
43     // check length conversion factors
44     cf = conversion_factor( HepMC::Units::MM, HepMC::Units::MM );
45     if( cf != 1 ) {
46         ++err;
47         std::cerr << "wrong conversion factor " << cf
48             << " for MM to MM - should be 1 \n";
49     }
50     cf = conversion_factor( HepMC::Units::CM, HepMC::Units::CM );
51     if( cf != 1 ) {
52         ++err;
53         std::cerr << "wrong conversion factor " << cf
54             << " for CM to CM - should be 1 \n";
55     }
56     cf = conversion_factor( HepMC::Units::CM, HepMC::Units::MM );
57     if( cf != 10.0 ) {
58         ++err;
59         std::cerr << "wrong conversion factor " << cf
60             << " for CM to MM - should be 10 \n";
61     }

```

```
62  cf = conversion_factor( HepMC::Units::MM, HepMC::Units::CM );
63  if( cf != 0.1 ) {
64      ++err;
65      std::cerr << "wrong conversion factor " << cf
66                << " for MM to CM - should be 0.1 \n";
67  }
68
69  return err;
70 }
```

10.19 VectorConversion.h

This example converts from ThreeVector and FourVector to CLHEP::Hep3Vector and CLHEP::HepLorentzVector. Similar (or perhaps templated) conversion methods could be added to any vector class.

```
1 #ifndef VECTOR_CONVERSION_H
2 #define VECTOR_CONVERSION_H
3 // garren@fnal.gov, January 2007
4 //
5 // This example converts from ThreeVector and FourVector to
6 // CLHEP::Hep3Vector and CLHEP::HepLorentzVector
7 // Similar (or perhaps templated) conversion methods could be added to
8 // any vector class.
9 //
10 //
11
12
13 #include "HepMC/SimpleVector.h"
14 #include "CLHEP/Vector/LorentzVector.h"
15
16
17 inline CLHEP::Hep3Vector convertTo( const HepMC::ThreeVector& v )
18     { return CLHEP::Hep3Vector( v.x(), v.y(), v.z() ); }
19
20
21 inline CLHEP::HepLorentzVector convertTo( const HepMC::FourVector& v )
22     { return CLHEP::HepLorentzVector( v.x(), v.y(), v.z(), v.t() ); }
23
24
25
26 #endif // VECTOR_CONVERSION_H
```

Index

- /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/HepMC/Map
 - Directory Reference, 13
 - HepMC::TempParticleMap, 236
- /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/WeightContainers/
 - Directory Reference, 11
 - HepMC::WeightContainer, 246
- /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/Vertex
 - Directory Reference, 12
 - HepMC::GenVertex::edge_
 - iterator, 124
 - Directory Reference, 14
 - ~particle_const_iterator
- /home/cepa01/garren/lcg/hepmc/HepMC-2.05.01/HepMC/GenEvent
 - Directory Reference, 15
 - ~particle_iterator
 - HepMC::GenEvent::particle_
 - const_iterator, 87
- ~Flow
 - HepMC::Flow, 45
- ~GenCrossSection
 - HepMC::GenCrossSection, 61
- ~GenEvent
 - HepMC::GenEvent, 70
- ~GenParticle
 - HepMC::GenParticle, 102
- ~GenVertex
 - HepMC::GenVertex, 113
- ~HeavyIon
 - HepMC::HeavyIon, 135
- ~IO_AsciiParticles
 - HepMC::IO_AsciiParticles, 155
- ~IO_BaseClass
 - HepMC::IO_BaseClass, 159
- ~IO_GenEvent
 - HepMC::IO_GenEvent, 166
- ~IO_HEPEVT
 - HepMC::IO_HEPEVT, 170
- ~IO_HERWIG
 - HepMC::IO_HERWIG, 175
- ~IO_PDG_ParticleDataTable
 - HepMC::IO_PDG_ParticleData-
 - Table, 182
- ~ParticleData
 - HepMC::ParticleData, 206
- ~ParticleDataTable
 - HepMC::ParticleDataTable, 213
- ~PdfInfo
 - HepMC::PdfInfo, 220
- ~Polarization
 - HepMC::Polarization, 226
- ~StreamInfo
 - HepMC::StreamInfo, 231
- ~particle_iterator
 - HepMC::GenEvent::particle_
 - iterator, 90
 - HepMC::GenVertex::particle_
 - iterator, 127
- ~vertex_const_iterator
 - HepMC::GenEvent::vertex_const_
 - iterator, 93
- ~vertex_iterator
 - HepMC::GenEvent::vertex_
 - iterator, 96
 - HepMC::GenVertex::vertex_
 - iterator, 130
- add_particle_in
 - HepMC::GenVertex, 113
- add_particle_out
 - HepMC::GenVertex, 113
- add_quarks_to_table
 - HepMC::IO_PDG_ParticleData-
 - Table, 182
- add_vertex
 - HepMC::GenEvent, 70
- addEndParticle
 - HepMC::TempParticleMap, 236
- advance_to_first_
 - HepMC::GenVertex::particle_
 - iterator, 127
- AFCH
 - HerwigWrapper6_4.h, 291
- alphaQCD
 - HepMC::GenEvent, 71
- alphaQED
 - HepMC::GenEvent, 71
- ALPHEM

- HerwigWrapper6_4.h, 291
- already_in_vector
 - HepMC, 27
- ancestors
 - HepMC, 22
- ascii
 - HepMC, 23
- ascii_pdt
 - HepMC, 23
- AVWGT
 - HerwigWrapper6_4.h, 292
- AZSOFT
 - HerwigWrapper6_4.h, 292
- AZSPIN
 - HerwigWrapper6_4.h, 292
- B1LIM
 - HerwigWrapper6_4.h, 292
- back
 - HepMC::WeightContainer, 246
- BadInputStream
 - HepMC::IO_Exception, 163
- BadOutputStream
 - HepMC::IO_Exception, 163
- barcode
 - HepMC::GenParticle, 102
 - HepMC::GenVertex, 114
- barcode_to_particle
 - HepMC::GenEvent, 71
- barcode_to_vertex
 - HepMC::GenEvent, 71
- beam_particles
 - HepMC::GenEvent, 72
- begin
 - HepMC::Flow, 45, 46
 - HepMC::ParticleDataTable, 213
 - HepMC::TempParticleMap, 236
 - HepMC::WeightContainer, 246
- BETAF
 - HerwigWrapper6_4.h, 292
- brat
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- BTCLM
 - HerwigWrapper6_4.h, 292
- build_end_vertex
 - HepMC::IO_HEPEVT, 170
 - HepMC::IO_HERWIG, 176
- build_particle
 - HepMC::IO_HEPEVT, 171
 - HepMC::IO_HERWIG, 176
- build_production_vertex
 - HepMC::IO_HEPEVT, 171
- HepMC::IO_HERWIG, 176
- byte_num_to_double
 - HepMC::HEPEVT_Wrapper, 144
- byte_num_to_int
 - HepMC::HEPEVT_Wrapper, 144
- CAFAC
 - HerwigWrapper6_4.h, 292
- CFFAC
 - HerwigWrapper6_4.h, 292
- change_parent_event
 - HepMC::GenVertex, 114
- charge
 - HepMC::ParticleData, 206
- check_hepevt_consistency
 - HepMC::HEPEVT_Wrapper, 144
- check_momentum_conservation
 - HepMC::GenVertex, 114
- children
 - HepMC, 22
- ckin
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- CLDIR
 - HerwigWrapper6_4.h, 292
- clear
 - HepMC::Flow, 46
 - HepMC::GenCrossSection, 61
 - HepMC::GenEvent, 72
 - HepMC::IO_AsciiParticles, 155
 - HepMC::IO_GenEvent, 166
 - HepMC::ParticleDataTable, 214
 - HepMC::WeightContainer, 247
- CLHEP, 17
- clifetime
 - HepMC::ParticleData, 206
- clifetime_from_width
 - HepMC, 27
- CLMAX
 - HerwigWrapper6_4.h, 292
- CLPOW
 - HerwigWrapper6_4.h, 292
- CLSMR
 - HerwigWrapper6_4.h, 292
- CM
 - HepMC::Units, 35
- coef
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- compareBeamParticles
 - HepMC, 23
- compareGenEvent

- HepMC, 23
- CompareGenEvent.cc, 251
- CompareGenEvent.h, 252
- compareParticles
 - HepMC, 24
- compareSignalProcessVertex
 - HepMC, 23
- compareVertex
 - HepMC, 24
- compareVertices
 - HepMC, 24
- compareWeights
 - HepMC, 23
- connected_partners
 - HepMC::Flow, 46
- const_iterator
 - HepMC::Flow, 45
 - HepMC::ParticleDataTable, 213
 - HepMC::WeightContainer, 245
- conversion_factor
 - HepMC::Units, 36
- convert_momentum
 - HepMC::GenParticle, 102
- convert_position
 - HepMC::GenVertex, 114
- convert_units
 - HepMC, 26
- convertTo
 - VectorConversion.h, 371
- copy_if
 - HepMC, 24
- copy_recursive_iterator_
 - HepMC::GenVertex::vertex_
 - iterator, 131
- copy_with_own_set
 - HepMC::GenVertex::vertex_
 - iterator, 131
- counter
 - HepMC::ParticleData, 207
- cross_section
 - HepMC::GenCrossSection, 61
 - HepMC::GenEvent, 72
- cross_section_error
 - HepMC::GenCrossSection, 61
- CSPEED
 - HerwigWrapper6_4.h, 293
- dangling_connected_partners
 - HepMC::Flow, 46
- data
 - HEPEVT_Wrapper.h, 280
- default_length_unit
 - HepMC::Units, 36
- default_momentum_unit
 - HepMC::Units, 36
- delete_adopted_particles
 - HepMC::GenVertex, 115
- delete_all
 - HepMC::ParticleDataTable, 214
- delete_all_vertices
 - HepMC::GenEvent, 72
- descendants
 - HepMC, 22
- description
 - HepMC::ParticleDataTable, 214
- detail, 18
- e
 - HepMC::FourVector, 53
 - HepMC::HEPEVT_Wrapper, 144
- EBEAM1
 - HerwigWrapper6_4.h, 293
- EBEAM2
 - HerwigWrapper6_4.h, 293
- eccentricity
 - HepMC::HeavyIon, 136
- edge_iterator
 - HepMC::GenVertex, 121
 - HepMC::GenVertex::edge_
 - iterator, 124
- edges_begin
 - HepMC::GenVertex, 115
- edges_end
 - HepMC::GenVertex, 115
- edges_size
 - HepMC::GenVertex, 115
- EFFMIN
 - HerwigWrapper6_4.h, 293
- empty
 - HepMC::Flow, 47
 - HepMC::ParticleDataTable, 214
 - HepMC::WeightContainer, 247
- enable_if.h, 253
- end
 - HepMC::Flow, 47
 - HepMC::ParticleDataTable, 214
 - HepMC::TempParticleMap, 236
 - HepMC::WeightContainer, 247
- end_vertex
 - HepMC::GenParticle, 102
 - HepMC::TempParticleMap, 236
- EndKeyMismatch
 - HepMC::IO_Exception, 163
- EndOfStream
 - HepMC::IO_Exception, 163
- ENSOF
 - HerwigWrapper6_4.h, 293
- erase

- HepMC::Flow, 47
- HepMC::ParticleDataTable, 214, 215
- error_message
 - HepMC::IO_GenEvent, 166
- error_type
 - HepMC::IO_GenEvent, 166
- ErrorType
 - HepMC::IO_Exception, 163
- establish_input_stream_info
 - HepMC, 29
 - HepMC::detail, 32
- establish_output_stream_info
 - HepMC, 29
 - HepMC::detail, 32
- ET2MIX
 - HerwigWrapper6_4.h, 293
- eta
 - HepMC::FourVector, 53
- ETAMIX
 - HerwigWrapper6_4.h, 293
- event_number
 - HepMC::GenEvent, 73
 - HepMC::HEPEVT_Wrapper, 144
- event_plane_angle
 - HepMC::HeavyIon, 136
- event_scale
 - HepMC::GenEvent, 73
- event_selection
 - example_MyPythia.cc, 257
- EVWGT
 - HerwigWrapper6_4.h, 293
- example_BuildEventFromScratch.cc, 254
- example_BuildEventFromScratch.cc
 - main, 254
- example_EventSelection.cc, 255
- example_EventSelection.cc
 - main, 255
- example_MyHerwig.cc, 256
- example_MyHerwig.cc
 - main, 256
- example_MyPythia.cc, 257
- example_MyPythia.cc
 - event_selection, 257
 - main, 257
 - pythia_in, 257
 - pythia_in_out, 258
 - pythia_out, 258
 - pythia_particle_out, 258
- example_MyPythiaOnlyToHepMC.cc, 260
- example_MyPythiaOnlyToHepMC.cc
 - main, 260
- example_PythiaStreamIO.cc, 261
- example_PythiaStreamIO.cc
 - main, 261
 - readPythiaStreamIO, 261
 - writePythiaStreamIO, 261
- example_UsingIterators.cc, 263
- example_UsingIterators.cc
 - main, 263
- extascii
 - HepMC, 23
- extascii_pdt
 - HepMC, 23
- F0MIX
 - HerwigWrapper6_4.h, 293
- F1MIX
 - HerwigWrapper6_4.h, 293
- F2MIX
 - HerwigWrapper6_4.h, 293
- family
 - HepMC, 22
- fill_next_event
 - HepMC::IO_AsciiParticles, 155
 - HepMC::IO_BaseClass, 159
 - HepMC::IO_GenEvent, 166
 - HepMC::IO_HEPEVT, 171
 - HepMC::IO_HERWIG, 176
- fill_particle_data_table
 - HepMC::IO_AsciiParticles, 155
 - HepMC::IO_BaseClass, 159
 - HepMC::IO_GenEvent, 166
 - HepMC::IO_PDG_ParticleData-Table, 182
- find
 - HepMC::ParticleDataTable, 215
- find_event_end
 - HepMC::detail, 34
- find_in_map
 - HepMC::IO_HEPEVT, 171
 - HepMC::IO_HERWIG, 177
- findPiZero
 - testHepMCMethods.cc, 363
 - testHepMCMethods.h, 364
- finished_first_event
 - HepMC::StreamInfo, 231
- first_child
 - HepMC::HEPEVT_Wrapper, 145
- first_parent
 - HepMC::HEPEVT_Wrapper, 145
- Flow
 - HepMC::Flow, 45
- flow
 - HepMC::GenParticle, 103
- Flow.cc, 264

- Flow.h, 265
- FlowVec
 - testFlow.cc, 361
- follow_edge_
 - HepMC::GenVertex::vertex_
 - iterator, 131
- FourVector
 - HepMC::FourVector, 52
- front
 - HepMC::WeightContainer, 247
- GAMH
 - HerwigWrapper6_4.h, 293
- GAMW
 - HerwigWrapper6_4.h, 294
- GAMWT
 - HerwigWrapper6_4.h, 294
- GAMZ
 - HerwigWrapper6_4.h, 294
- GAMZP
 - HerwigWrapper6_4.h, 294
- GCUTME
 - HerwigWrapper6_4.h, 294
- gen
 - HepMC, 23
- GenCrossSection
 - HepMC::GenCrossSection, 61
- GenCrossSection.cc, 266
- GenCrossSection.h, 267
- generated_mass
 - HepMC::GenParticle, 103
- generatedMass
 - HepMC::GenParticle, 103
- GenEvent
 - HepMC::GenEvent, 69, 70
 - HepMC::GenParticle, 108
 - HepMC::GenVertex, 121
- GenEvent.cc, 268
- GenEvent.h, 269
- GenEventStreamIO.cc, 271
- GenParticle
 - HepMC::GenEvent, 84
 - HepMC::GenParticle, 101, 102
- GenParticle.cc, 273
- GenParticle.h, 274
- GenParticle.h
 - hepmc_uint64_t, 274
- GENSOF
 - HerwigWrapper6_4.h, 294
- GenVertex
 - HepMC::GenEvent, 84
 - HepMC::GenParticle, 108
 - HepMC::GenVertex, 113
- GenVertex.cc, 275
- GenVertex.h, 276
- get_stream_info
 - HepMC, 29
- getHerwigCrossSection
 - HerwigHelper.h, 283
- getPythiaCrossSection
 - PythiaHelper.h, 326
- GEV
 - HepMC::Units, 35
- GEV2NB
 - HerwigWrapper6_4.h, 294
- H1MIX
 - HerwigWrapper6_4.h, 294
- HARDME
 - HerwigWrapper6_4.h, 294
- has_decayed
 - HepMC::GenParticle, 103
- has_key
 - HepMC::StreamInfo, 231
- heavy_ion
 - HepMC::GenEvent, 73
- HeavyIon
 - HepMC::HeavyIon, 135
- HeavyIon.cc, 277
- HeavyIon.h, 278
- hepevt
 - HEPEVT_Wrapper.h, 280
- hepevt_
 - HEPEVT_Wrapper.h, 280
- hepevt_bytes_allocation
 - HEPEVT_Wrapper.h, 281
- HEPEVT_EntriesAllocation
 - HEPEVT_Wrapper.h, 280
- HEPEVT_Wrapper.cc, 279
- HEPEVT_Wrapper.h, 280
 - data, 280
 - hepevt, 280
 - hepevt_, 280
 - hepevt_bytes_allocation, 281
 - HEPEVT_EntriesAllocation, 280
- HepMC, 19
 - ancestors, 22
 - ascii, 23
 - ascii_pdt, 23
 - children, 22
 - descendants, 22
 - extascii, 23
 - extascii_pdt, 23
 - family, 22
 - gen, 23
 - parents, 22
 - relatives, 23
- HepMC

- already_in_vector, 27
- clifetime_from_width, 27
- compareBeamParticles, 23
- compareGenEvent, 23
- compareParticles, 24
- compareSignalProcessVertex, 23
- compareVertex, 24
- compareVertices, 24
- compareWeights, 23
- convert_units, 26
- copy_if, 24
- establish_input_stream_info, 29
- establish_output_stream_info, 29
- get_stream_info, 29
- HepMC_hbarc, 30
- HepMC_pi, 30
- HepMCStreamCallback, 28
- IteratorRange, 22
- known_io, 23
- not_in_vector, 27
- operator<<, 24-30
- operator>>, 24-27
- set_input_units, 25
- version, 28
- versionName, 28
- write_HepMC_IO_block_begin, 25
- write_HepMC_IO_block_end, 26
- writeVersion, 28
- HepMC::detail, 31
- HepMC::detail
 - establish_input_stream_info, 32
 - establish_output_stream_info, 32
 - find_event_end, 34
 - output, 33, 34
 - read_particle, 33
 - read_units, 34
 - read_vertex, 33
- HepMC::detail::disable_if, 39
- HepMC::detail::disable_if< false, T >, 40
- HepMC::detail::disable_if< false, T >
 - type, 40
- HepMC::detail::enable_if, 41
- HepMC::detail::enable_if< true, T >, 42
- HepMC::detail::enable_if< true, T >
 - type, 42
- HepMC::detail::is_arithmetic, 184
- HepMC::detail::is_arithmetic
 - value, 184
- HepMC::detail::is_arithmetic< char >, 185
- HepMC::detail::is_arithmetic< char >
 - value, 185
- HepMC::detail::is_arithmetic< double >, 186
- HepMC::detail::is_arithmetic< double >
 - value, 186
- HepMC::detail::is_arithmetic< float >, 187
- HepMC::detail::is_arithmetic< float >
 - value, 187
- HepMC::detail::is_arithmetic< int >, 188
- HepMC::detail::is_arithmetic< int >
 - value, 188
- HepMC::detail::is_arithmetic< long >, 189
- HepMC::detail::is_arithmetic< long >
 - value, 189
- HepMC::detail::is_arithmetic< long double >, 190
- HepMC::detail::is_arithmetic< long double >
 - value, 190
- HepMC::detail::is_arithmetic< short >, 191
- HepMC::detail::is_arithmetic< short >
 - value, 191
- HepMC::detail::is_arithmetic< signed char >, 192
- HepMC::detail::is_arithmetic< signed char >
 - value, 192
- HepMC::detail::is_arithmetic< unsigned char >, 193
- HepMC::detail::is_arithmetic< unsigned char >
 - value, 193
- HepMC::detail::is_arithmetic< unsigned int >, 194
- HepMC::detail::is_arithmetic< unsigned int >
 - value, 194
- HepMC::detail::is_arithmetic< unsigned long >, 195
- HepMC::detail::is_arithmetic< unsigned long >

- value, 195
- HepMC::detail::is_arithmetic<
 - unsigned short >, 196
- HepMC::detail::is_arithmetic<
 - unsigned short >
 - value, 196
- HepMC::Flow, 43
- HepMC::Flow
 - ~Flow, 45
 - begin, 45, 46
 - clear, 46
 - connected_partners, 46
 - const_iterator, 45
 - dangling_connected_partners, 46
 - empty, 47
 - end, 47
 - erase, 47
 - Flow, 45
 - icode, 47
 - iterator, 45
 - operator!=, 47
 - operator<=, 49
 - operator=, 47
 - operator==, 48
 - particle_owner, 48
 - print, 48
 - set_icode, 48
 - set_unique_icode, 48
 - size, 48
 - swap, 48
- HepMC::FourVector, 50
- HepMC::FourVector
 - e, 53
 - eta, 53
 - FourVector, 52
 - m, 53
 - m2, 53
 - operator!=, 53
 - operator=, 54
 - operator==, 54
 - perp, 54
 - perp2, 54
 - phi, 54
 - pseudoRapidity, 54
 - px, 54
 - py, 55
 - pz, 55
 - rho, 55
 - set, 55
 - setE, 56
 - setPx, 56
 - setPy, 56
 - setPz, 56
 - setT, 56
 - setX, 57
 - setY, 57
 - setZ, 57
 - swap, 57
 - t, 57
 - theta, 58
 - x, 58
 - y, 58
 - z, 58
- HepMC::GenCrossSection, 60
- HepMC::GenCrossSection
 - ~GenCrossSection, 61
 - clear, 61
 - cross_section, 61
 - cross_section_error, 61
 - GenCrossSection, 61
 - is_set, 62
 - operator!=, 62
 - operator=, 62
 - operator==, 62
 - read, 62
 - set_cross_section, 62
 - set_cross_section_error, 63
 - swap, 63
 - write, 63
- HepMC::GenEvent, 64
- HepMC::GenEvent
 - ~GenEvent, 70
 - add_vertex, 70
 - alphaQCD, 71
 - alphaQED, 71
 - barcode_to_particle, 71
 - barcode_to_vertex, 71
 - beam_particles, 72
 - clear, 72
 - cross_section, 72
 - delete_all_vertices, 72
 - event_number, 73
 - event_scale, 73
 - GenEvent, 69, 70
 - GenParticle, 84
 - GenVertex, 84
 - heavy_ion, 73
 - is_valid, 73
 - length_unit, 74
 - momentum_unit, 74
 - mpi, 74
 - operator=, 74
 - particle_const_iterator, 84
 - particle_iterator, 84
 - particles_begin, 74
 - particles_empty, 75
 - particles_end, 75
 - particles_size, 75

```

pdf_info, 75, 76
print, 76
print_version, 76
random_states, 76
read, 76
remove_barcode, 77
remove_vertex, 77
set_alphaQCD, 77
set_alphaQED, 78
set_barcode, 78
set_beam_particles, 78
set_cross_section, 78
set_event_number, 79
set_event_scale, 79
set_heavy_ion, 79
set_mpi, 79
set_pdf_info, 80
set_random_states, 80
set_signal_process_id, 80
set_signal_process_vertex, 80
signal_process_id, 80
signal_process_vertex, 81
swap, 81
use_units, 81
valid_beam_particles, 81
vertex_const_iterator, 84
vertex_iterator, 85
vertices_begin, 82
vertices_empty, 82
vertices_end, 82
vertices_size, 83
weights, 83
write, 83
write_cross_section, 83
write_units, 84
HepMC::GenEvent::particle_const_
    iterator, 86
HepMC::GenEvent::particle_const_
    iterator
    ~particle_const_iterator, 87
    m_map_iterator, 88
    operator *, 87
    operator !=, 87
    operator ++, 87, 88
    operator =, 88
    operator ==, 88
    particle_const_iterator, 87
HepMC::GenEvent::particle_
    iterator, 89
HepMC::GenEvent::particle_iterator
    ~particle_iterator, 90
    m_map_iterator, 91
    operator *, 90
    operator particle_const_
        iterator, 90
    operator !=, 90
    operator ++, 91
    operator =, 91
    operator ==, 91
    particle_iterator, 90
HepMC::GenEvent::vertex_const_
    iterator, 92
HepMC::GenEvent::vertex_const_
    iterator
    ~vertex_const_iterator, 93
    m_map_iterator, 94
    operator *, 93
    operator !=, 93
    operator ++, 93
    operator =, 94
    operator ==, 94
    vertex_const_iterator, 93
HepMC::GenEvent::vertex_iterator,
    95
HepMC::GenEvent::vertex_iterator
    ~vertex_iterator, 96
    m_map_iterator, 97
    operator *, 96
    operator vertex_const_iterator,
        96
    operator !=, 97
    operator ++, 97
    operator =, 97
    operator ==, 97
    vertex_iterator, 96
HepMC::GenParticle, 99
HepMC::GenParticle
    ~GenParticle, 102
    barcode, 102
    convert_momentum, 102
    end_vertex, 102
    flow, 103
    generated_mass, 103
    generatedMass, 103
    GenEvent, 108
    GenParticle, 101, 102
    GenVertex, 108
    has_decayed, 103
    is_beam, 103
    is_undecayed, 104
    momentum, 104
    operator HepMC::FourVector, 104
    operator !=, 104
    operator <<, 108
    operator =, 104
    operator ==, 104
    parent_event, 104

```

- pdg_id, 105
- polarization, 105
- print, 105
- production_vertex, 105
- set_barcode_, 105
- set_end_vertex_, 105
- set_flow, 106
- set_generated_mass, 106
- set_momentum, 106
- set_pdg_id, 106
- set_polarization, 106
- set_production_vertex_, 107
- set_status, 107
- setGeneratedMass, 107
- status, 107
- suggest_barcode, 107
- swap, 107
- HepMC::GenVertex, 109
- HepMC::GenVertex
 - ~GenVertex, 113
 - add_particle_in, 113
 - add_particle_out, 113
 - barcode, 114
 - change_parent_event_, 114
 - check_momentum_conservation, 114
 - convert_position, 114
 - delete_adopted_particles, 115
 - edge_iterator, 121
 - edges_begin, 115
 - edges_end, 115
 - edges_size, 115
 - GenEvent, 121
 - GenVertex, 113
 - id, 115
 - operator HepMC::FourVector, 115
 - operator HepMC::ThreeVector, 116
 - operator!=, 116
 - operator<<, 121
 - operator=, 116
 - operator==, 116
 - parent_event, 116
 - particle_iterator, 121
 - particles_begin, 117
 - particles_end, 117
 - particles_in_const_begin, 117
 - particles_in_const_end, 117
 - particles_in_const_iterator, 112
 - particles_in_size, 117
 - particles_out_const_begin, 117
 - particles_out_const_end, 118
 - particles_out_const_iterator, 112
 - particles_out_size, 118
 - point3d, 118
 - position, 118
 - print, 118
 - remove_particle, 119
 - remove_particle_in, 119
 - remove_particle_out, 119
 - set_barcode_, 119
 - set_id, 119
 - set_parent_event_, 120
 - set_position, 120
 - suggest_barcode, 120
 - swap, 120
 - vertex_iterator, 122
 - vertices_begin, 121
 - vertices_end, 121
 - weights, 121
- HepMC::GenVertex::edge_iterator, 123
- HepMC::GenVertex::edge_iterator
 - ~edge_iterator, 124
 - edge_iterator, 124
 - is_child, 124
 - is_parent, 124
 - operator *, 124
 - operator!=, 124
 - operator++, 125
 - operator=, 125
 - operator==, 125
 - vertex_root, 125
- HepMC::GenVertex::particle_iterator, 126
- HepMC::GenVertex::particle_iterator
 - ~particle_iterator, 127
 - advance_to_first_, 127
 - operator *, 127
 - operator!=, 127
 - operator++, 128
 - operator=, 128
 - operator==, 128
 - particle_iterator, 127
- HepMC::GenVertex::vertex_iterator, 129
- HepMC::GenVertex::vertex_iterator
 - ~vertex_iterator, 130
 - copy_recursive_iterator_, 131
 - copy_with_own_set, 131
 - follow_edge_, 131
 - operator *, 131
 - operator!=, 131
 - operator++, 131, 132

- operator=, 132
- operator==, 132
- range, 132
- vertex_iterator, 130
- vertex_root, 132
- HepMC::HeavyIon, 133
- HepMC::HeavyIon
 - ~HeavyIon, 135
 - eccentricity, 136
 - event_plane_angle, 136
 - HeavyIon, 135
 - impact_parameter, 136
 - is_valid, 136
 - N_Nwounded_collisions, 136
 - Ncoll, 136
 - Ncoll_hard, 136
 - Npart_proj, 136
 - Npart_targ, 137
 - Nwounded_N_collisions, 137
 - Nwounded_Nwounded_collisions, 137
 - operator!=, 137
 - operator=, 137
 - operator==, 137
 - set_eccentricity, 138
 - set_event_plane_angle, 138
 - set_impact_parameter, 138
 - set_N_Nwounded_collisions, 138
 - set_Ncoll, 138
 - set_Ncoll_hard, 138
 - set_Npart_proj, 138
 - set_Npart_targ, 139
 - set_Nwounded_N_collisions, 139
 - set_Nwounded_Nwounded_collisions, 139
 - set_sigma_inel_NN, 139
 - set_spectator_neutrons, 139
 - set_spectator_protons, 139
 - sigma_inel_NN, 139
 - spectator_neutrons, 140
 - spectator_protons, 140
 - swap, 140
- HepMC::HEPEVT_Wrapper, 141
- HepMC::HEPEVT_Wrapper
 - byte_num_to_double, 144
 - byte_num_to_int, 144
 - check_hepevt_consistency, 144
 - e, 144
 - event_number, 144
 - first_child, 145
 - first_parent, 145
 - id, 145
 - is_double_precision, 145
 - last_child, 145
 - last_parent, 146
 - m, 146
 - max_number_entries, 146
 - number_children, 146
 - number_entries, 146
 - number_parents, 147
 - print_hepevt, 147
 - print_hepevt_particle, 147
 - print_legend, 147
 - px, 147
 - py, 148
 - pz, 148
 - set_children, 148
 - set_event_number, 148
 - set_id, 148
 - set_mass, 149
 - set_max_number_entries, 149
 - set_momentum, 149
 - set_number_entries, 149
 - set_parents, 150
 - set_position, 150
 - set_sizeof_int, 150
 - set_sizeof_real, 150
 - set_status, 150
 - sizeof_int, 151
 - sizeof_real, 151
 - status, 151
 - t, 151
 - write_byte_num, 151
 - x, 152
 - y, 152
 - z, 152
 - zero_everything, 152
- HepMC::IO_AsciiParticles, 154
- HepMC::IO_AsciiParticles
 - ~IO_AsciiParticles, 155
 - clear, 155
 - fill_next_event, 155
 - fill_particle_data_table, 155
 - IO_AsciiParticles, 155
 - print, 155
 - rdstate, 156
 - setPrecision, 156
 - write_comment, 156
 - write_end_listing, 156
 - write_event, 156
 - write_particle_data_table, 156
- HepMC::IO_BaseClass, 158
- HepMC::IO_BaseClass
 - ~IO_BaseClass, 159
 - fill_next_event, 159
 - fill_particle_data_table, 159
 - operator<<, 159
 - operator>>, 160

- print, 160
- read_next_event, 160
- read_particle_data_table, 160
- write_event, 161
- write_particle_data_table, 161
- HepMC::IO_Exception, 162
 - BadInputStream, 163
 - BadOutputStream, 163
 - EndKeyMismatch, 163
 - EndOfStream, 163
 - InputAndOutput, 163
 - InvalidData, 163
 - MissingEndKey, 163
 - MissingStartKey, 163
 - NullEvent, 163
 - OK, 163
 - WrongFileType, 163
- HepMC::IO_Exception
 - ErrorType, 163
 - IO_Exception, 163
- HepMC::IO_GenEvent, 164
- HepMC::IO_GenEvent
 - ~IO_GenEvent, 166
 - clear, 166
 - error_message, 166
 - error_type, 166
 - fill_next_event, 166
 - fill_particle_data_table, 166
 - IO_GenEvent, 165, 166
 - precision, 167
 - print, 167
 - rdstate, 167
 - read_io_particle_data_table, 167
 - read_particle_data, 167
 - use_input_units, 167
 - write_comment, 168
 - write_event, 168
 - write_particle_data_table, 168
- HepMC::IO_HEPEVT, 169
- HepMC::IO_HEPEVT
 - ~IO_HEPEVT, 170
 - build_end_vertex, 170
 - build_particle, 171
 - build_production_vertex, 171
 - fill_next_event, 171
 - find_in_map, 171
 - IO_HEPEVT, 170
 - print, 172
 - print_inconsistency_errors, 172
 - set_print_inconsistency_errors, 172
 - set_trust_beam_particles, 172
 - set_trust_both_mothers_and_daughters, 172
 - set_trust_mothers_before_daughters, 173
 - trust_beam_particles, 173
 - trust_both_mothers_and_daughters, 173
 - trust_mothers_before_daughters, 173
 - write_event, 173
- HepMC::IO_HERWIG, 174
- HepMC::IO_HERWIG
 - ~IO_HERWIG, 175
 - build_end_vertex, 176
 - build_particle, 176
 - build_production_vertex, 176
 - fill_next_event, 176
 - find_in_map, 177
 - interfaces_to_version_number, 177
 - IO_HERWIG, 175
 - no_gaps_in_barcodes, 177
 - print, 177
 - print_inconsistency_errors, 177
 - remove_gaps_in_hepevt, 177
 - repair_hepevt, 178
 - set_no_gaps_in_barcodes, 178
 - set_print_inconsistency_errors, 179
 - set_trust_both_mothers_and_daughters, 179
 - set_trust_mothers_before_daughters, 179
 - translate_herwig_to_pdg_id, 179
 - trust_both_mothers_and_daughters, 179
 - trust_mothers_before_daughters, 179
 - zero_hepevt_entry, 179
- HepMC::IO_PDG_ParticleDataTable, 181
- HepMC::IO_PDG_ParticleDataTable
 - ~IO_PDG_ParticleDataTable, 182
 - add_quarks_to_table, 182
 - fill_particle_data_table, 182
 - IO_PDG_ParticleDataTable, 182
 - print, 182
 - rdstate, 182
 - read_entry, 182
 - search_for_key_end, 183
- HepMC::ParticleData, 204
- HepMC::ParticleData
 - ~ParticleData, 206
 - charge, 206

clifetime, 206
counter, 207
is_boson, 207
is_charged_lepton, 207
is_em, 207
is_hadron, 207
is_lepton, 207
is_neutrino, 208
mass, 208
model_independent_pdg_id_, 208
name, 208
operator!=, 208
operator<<, 210
operator==, 208
ParticleData, 206
pdg_id, 209
print, 209
set_charge, 209
set_clifetime, 209
set_mass, 209
set_spin, 209
set_width, 209
spin, 210
width, 210
HepMC::ParticleDataTable, 211
HepMC::ParticleDataTable
 ~ParticleDataTable, 213
 begin, 213
 clear, 214
 const_iterator, 213
 delete_all, 214
 description, 214
 empty, 214
 end, 214
 erase, 214, 215
 find, 215
 insert, 215
 iterator, 213
 make_antiparticles_from_
 particles, 215
 merge_table, 215
 operator=, 216
 operator[], 216
 ParticleDataTable, 213
 print, 216
 set_description, 216
 size, 217
HepMC::PdfInfo, 218
HepMC::PdfInfo
 ~PdfInfo, 220
 id1, 220
 id2, 220
 is_valid, 221
 operator!=, 221
 operator=, 221
 operator==, 221
 pdf1, 221
 pdf2, 221
 pdf_id1, 221
 pdf_id2, 222
 PdfInfo, 220
 scalePDF, 222
 set_id1, 222
 set_id2, 222
 set_pdf1, 222
 set_pdf2, 222
 set_pdf_id1, 222
 set_pdf_id2, 223
 set_scalePDF, 223
 set_x1, 223
 set_x2, 223
 swap, 223
 x1, 223
 x2, 223
HepMC::Polarization, 225
HepMC::Polarization
 ~Polarization, 226
 normal3d, 227
 operator!=, 227
 operator<<, 228
 operator=, 227
 operator==, 227
 phi, 227
 Polarization, 226
 print, 227
 set_normal3d, 227
 set_phi, 227
 set_theta, 228
 set_theta_phi, 228
 swap, 228
 theta, 228
HepMC::StreamInfo, 230
HepMC::StreamInfo
 ~StreamInfo, 231
 finished_first_event, 231
 has_key, 231
 IO_Ascii_End, 232
 IO_Ascii_Key, 232
 IO_Ascii_PDT_End, 232
 IO_Ascii_PDT_Key, 232
 IO_ExtendedAscii_End, 232
 IO_ExtendedAscii_Key, 232
 IO_ExtendedAscii_PDT_End, 232
 IO_ExtendedAscii_PDT_Key, 232
 IO_GenEvent_End, 233
 IO_GenEvent_Key, 233
 io_momentum_unit, 233
 io_position_unit, 233

- io_type, 233
- set_finished_first_event, 233
- set_has_key, 233
- set_io_type, 234
- stream_id, 234
- StreamInfo, 231
- use_input_units, 234
- HepMC::TempParticleMap, 235
- HepMC::TempParticleMap
 - ~TempParticleMap, 236
 - addEndParticle, 236
 - begin, 236
 - end, 236
 - end_vertex, 236
 - order_begin, 236
 - order_end, 236
 - orderIterator, 235
 - TempMap, 235
 - TempMapIterator, 235
 - TempOrderMap, 235
 - TempParticleMap, 236
- HepMC::ThreeVector, 238
- HepMC::ThreeVector
 - operator!=, 240
 - operator=, 240
 - operator==, 240
 - perp, 240
 - perp2, 240
 - phi, 240
 - r, 241
 - set, 241
 - setPhi, 241
 - setTheta, 241
 - setX, 241
 - setY, 242
 - setZ, 242
 - swap, 242
 - theta, 242
 - ThreeVector, 239, 240
 - x, 242
 - y, 243
 - z, 243
- HepMC::Units, 35
 - CM, 35
 - GEV, 35
 - MEV, 35
 - MM, 35
- HepMC::Units
 - conversion_factor, 36
 - default_length_unit, 36
 - default_momentum_unit, 36
 - LengthUnit, 35
 - MomentumUnit, 35
 - name, 36
- HepMC::WeightContainer, 244
- HepMC::WeightContainer
 - ~WeightContainer, 246
 - back, 246
 - begin, 246
 - clear, 247
 - const_iterator, 245
 - empty, 247
 - end, 247
 - front, 247
 - iterator, 245
 - operator=, 247, 248
 - operator[], 248
 - pop_back, 248
 - print, 248
 - push_back, 248
 - size, 248
 - swap, 248
 - WeightContainer, 246
- HepMC_hbarc
 - HepMC, 30
- HepMC_pi
 - HepMC, 30
- hepmc_uint64_t
 - GenParticle.h, 274
- HEPMC_VERSION
 - HepMCDefs.h, 282
- HepMCDefs.h, 282
- HepMCDefs.h
 - HEPMC_VERSION, 282
- HepMCStreamCallback
 - HepMC, 28
- herwig_hepevt_size
 - HerwigWrapper6_4.h, 294
- HerwigHelper.h, 283
- HerwigHelper.h
 - getHerwigCrossSection, 283
- HerwigWrapper.h, 284
- HerwigWrapper6_4.h, 285
- HerwigWrapper6_4.h
 - AFCH, 291
 - ALPHEM, 291
 - AVWGT, 292
 - AZSOFT, 292
 - AZSPIN, 292
 - B1LIM, 292
 - BETAF, 292
 - BTCLM, 292
 - CAFAC, 292
 - CFFAC, 292
 - CLDIR, 292
 - CLMAX, 292
 - CLPOW, 292
 - CLSMR, 292

CSPEED, 293
EBEAM1, 293
EBEAM2, 293
EFFMIN, 293
ENSOF, 293
ET2MIX, 293
ETAMIX, 293
EVWGT, 293
F0MIX, 293
F1MIX, 293
F2MIX, 293
GAMH, 293
GAMW, 294
GAMWT, 294
GAMZ, 294
GAMZP, 294
GCUTME, 294
GENSOF, 294
GEV2NB, 294
H1MIX, 294
HARDME, 294
herwig_hepevt_size, 294
hwbgen, 288
hwbmch, 288
hwbmch_, 294
hwcdec, 288
hwcfor, 288
hwdhad, 288
hwdhob, 288
hwdhvy, 289
hwefin, 289
hwegup, 289
hweini, 289
hwepro, 289
hwevnt, 289
hwevnt_, 295
hwigin, 290
hwigup, 290
hwmevt, 290
hwpram, 290
hwpram_, 295
hwproc, 290
hwproc_, 295
hwudat, 290
hwudpr, 290
hwuepr, 290
hwufne, 291
hwuinc, 291
hwuine, 291
hwupro, 291
hwupup, 291
hwusta, 291
IDHW, 295
IERROR, 295
IOP4JT, 295
IOPREM, 295
IPRINT, 295
IPROC, 295
ISPAC, 295
ISTAT, 295
LRSUD, 295
LWEVT, 295
LWSUD, 295
MAXER, 296
MAXEV, 296
MAXPR, 296
MODPDF, 296
NBTRY, 296
NCOLO, 296
NCTRY, 296
NDTRY, 296
NETRY, 296
NFLAV, 296
NGSPL, 296
NOSPAC, 296
NOWGT, 297
NPRFMT, 297
NRN, 297
NSTRU, 297
NSTRY, 297
NUMER, 297
NUMERU, 297
NWGTS, 297
NZBIN, 297
OMHMIX, 297
PART1, 297
PART2, 297
PBEAM1, 298
PBEAM2, 298
PDIQK, 298
PGSMX, 298
PGSPL, 298
PH3MIX, 298
PHIMIX, 298
PIFAC, 298
PRNDEC, 298
PRNDEF, 298
PRNTEX, 298
PRNWEB, 298
PRSOF, 299
PRVTX, 299
PSPLT, 299
PTRMS, 299
PXRMS, 299
QCDL3, 299
QCDL5, 299
QCDLAM, 299
QDIQK, 299

- QFCH, 299
- QG, 299
- QSPAC, 299
- QV, 300
- SCABI, 300
- SOFTME, 300
- SWEIN, 300
- TLOUT, 300
- TMTOP, 300
- VCKM, 300
- VFCH, 300
- VGCUT, 300
- VPCUT, 300
- VQCUT, 300
- WBGST, 300
- WGTMX, 301
- WGTSUM, 301
- WSQSUM, 301
- ZBINM, 301
- ZPRIME, 301
- hwbgen
 - HerwigWrapper6_4.h, 288
- hwbmch
 - HerwigWrapper6_4.h, 288
- hwbmch_
 - HerwigWrapper6_4.h, 294
- hwcdec
 - HerwigWrapper6_4.h, 288
- hwcfor
 - HerwigWrapper6_4.h, 288
- hwdhad
 - HerwigWrapper6_4.h, 288
- hwdhob
 - HerwigWrapper6_4.h, 288
- hwdhvy
 - HerwigWrapper6_4.h, 289
- hwefin
 - HerwigWrapper6_4.h, 289
- hwegup
 - HerwigWrapper6_4.h, 289
- hweini
 - HerwigWrapper6_4.h, 289
- hwepro
 - HerwigWrapper6_4.h, 289
- hwevnt
 - HerwigWrapper6_4.h, 289
- hwevnt_
 - HerwigWrapper6_4.h, 295
- hwigin
 - HerwigWrapper6_4.h, 290
- hwigup
 - HerwigWrapper6_4.h, 290
- hwmevt
 - HerwigWrapper6_4.h, 290
- hwpram
 - HerwigWrapper6_4.h, 290
- hwpram_
 - HerwigWrapper6_4.h, 295
- hwproc
 - HerwigWrapper6_4.h, 290
- hwproc_
 - HerwigWrapper6_4.h, 295
- hwudat
 - HerwigWrapper6_4.h, 290
- hwudpr
 - HerwigWrapper6_4.h, 290
- hwuepr
 - HerwigWrapper6_4.h, 290
- hwufne
 - HerwigWrapper6_4.h, 291
- hwuinc
 - HerwigWrapper6_4.h, 291
- hwuine
 - HerwigWrapper6_4.h, 291
- hwupro
 - HerwigWrapper6_4.h, 291
- hwupup
 - HerwigWrapper6_4.h, 291
- hwusta
 - HerwigWrapper6_4.h, 291
- icode
 - HepMC::Flow, 47
- icol
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- id
 - HepMC::GenVertex, 115
 - HepMC::HEPEVT_Wrapper, 145
- idl
 - HepMC::PdfInfo, 220
- id2
 - HepMC::PdfInfo, 220
- IDHW
 - HerwigWrapper6_4.h, 295
- IERROR
 - HerwigWrapper6_4.h, 295
- impact_parameter
 - HepMC::HeavyIon, 136
- initpydata
 - PythiaWrapper5_720.h, 329
 - PythiaWrapper6_152.h, 337
 - PythiaWrapper6_2.h, 346, 347
- initPythia
 - initPythia.cc, 302
 - PythiaHelper.h, 326
- initPythia.cc, 302

initPythia.cc
 initPythia, 302
 InputAndOutput
 HepMC::IO_Exception, 163
 insert
 HepMC::ParticleDataTable, 215
 interfaces_to_version_number
 HepMC::IO_HERWIG, 177
 InvalidData
 HepMC::IO_Exception, 163
 IO_Ascii_End
 HepMC::StreamInfo, 232
 IO_Ascii_Key
 HepMC::StreamInfo, 232
 IO_Ascii_PDT_End
 HepMC::StreamInfo, 232
 IO_Ascii_PDT_Key
 HepMC::StreamInfo, 232
 IO_AsciiParticles
 HepMC::IO_AsciiParticles, 155
 IO_AsciiParticles.cc, 303
 IO_AsciiParticles.h, 304
 IO_BaseClass.h, 305
 IO_Exception
 HepMC::IO_Exception, 163
 IO_Exception.h, 306
 IO_ExtendedAscii_End
 HepMC::StreamInfo, 232
 IO_ExtendedAscii_Key
 HepMC::StreamInfo, 232
 IO_ExtendedAscii_PDT_End
 HepMC::StreamInfo, 232
 IO_ExtendedAscii_PDT_Key
 HepMC::StreamInfo, 232
 IO_GenEvent
 HepMC::IO_GenEvent, 165, 166
 IO_GenEvent.cc, 307
 IO_GenEvent.h, 308
 IO_GenEvent_End
 HepMC::StreamInfo, 233
 IO_GenEvent_Key
 HepMC::StreamInfo, 233
 IO_HEPEVT
 HepMC::IO_HEPEVT, 170
 IO_HEPEVT.cc, 309
 IO_HEPEVT.h, 310
 IO_HERWIG
 HepMC::IO_HERWIG, 175
 IO_HERWIG.cc, 311
 IO_HERWIG.h, 312
 io_momentum_unit
 HepMC::StreamInfo, 233
 IO_PDG_ParticleDataTable
 HepMC::IO_PDG_ParticleData-
 Table, 182
 IO_PDG_ParticleDataTable.cc, 313
 IO_PDG_ParticleDataTable.h, 314
 io_position_unit
 HepMC::StreamInfo, 233
 io_type
 HepMC::StreamInfo, 233
 IOP4JT
 HerwigWrapper6_4.h, 295
 IOPREM
 HerwigWrapper6_4.h, 295
 IPRINT
 HerwigWrapper6_4.h, 295
 IPROC
 HerwigWrapper6_4.h, 295
 is_arithmetic.h, 315
 is_beam
 HepMC::GenParticle, 103
 is_boson
 HepMC::ParticleData, 207
 is_charged_lepton
 HepMC::ParticleData, 207
 is_child
 HepMC::GenVertex::edge_
 iterator, 124
 is_double_precision
 HepMC::HEPEVT_Wrapper, 145
 is_em
 HepMC::ParticleData, 207
 is_hadron
 HepMC::ParticleData, 207
 is_lepton
 HepMC::ParticleData, 207
 is_neutrino
 HepMC::ParticleData, 208
 is_parent
 HepMC::GenVertex::edge_
 iterator, 124
 is_set
 HepMC::GenCrossSection, 62
 is_undecayed
 HepMC::GenParticle, 104
 is_valid
 HepMC::GenEvent, 73
 HepMC::HeavyIon, 136
 HepMC::PdfInfo, 221
 iset
 PythiaWrapper5_720.h, 331
 PythiaWrapper6_152.h, 339
 PythiaWrapper6_2.h, 348
 IsEventGood, 197
 IsEventGood
 operator(), 197

- IsFinalState, 198
- IsFinalState
 - operator(), 198
- IsGoodEvent, 199
- IsGoodEvent
 - operator(), 199
- IsGoodEvent.h, 316
- IsGoodEventMyPythia, 200
- IsGoodEventMyPythia
 - operator(), 200
- ISPAC
 - HerwigWrapper6_4.h, 295
- IsPhoton, 201
 - testHepMCIteration.h, 362
- IsPhoton
 - operator(), 201
- IsStateFinal, 202
- IsStateFinal
 - operator(), 202
- ISTAT
 - HerwigWrapper6_4.h, 295
- IsW_Boson, 203
- IsW_Boson
 - operator(), 203
- IsWBoson
 - testHepMCIteration.h, 362
- iterator
 - HepMC::Flow, 45
 - HepMC::ParticleDataTable, 213
 - HepMC::WeightContainer, 245
- IteratorRange
 - HepMC, 22
- k
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 348
- kchg
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 348
- kfdp
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 348
- kfin
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- kfpr
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- known_io
 - HepMC, 23
- last_child
 - HepMC::HEPEVT_Wrapper, 145
- last_parent
 - HepMC::HEPEVT_Wrapper, 146
- length_unit
 - HepMC::GenEvent, 74
- LengthUnit
 - HepMC::Units, 35
- list_of_examples.cc, 317, 318
- LRSUD
 - HerwigWrapper6_4.h, 295
- ludat1
 - PythiaWrapper5_720.h, 329
- ludat1_
 - PythiaWrapper5_720.h, 332
- ludat2
 - PythiaWrapper5_720.h, 330
- ludat2_
 - PythiaWrapper5_720.h, 332
- ludat3
 - PythiaWrapper5_720.h, 330
- ludat3_
 - PythiaWrapper5_720.h, 332
- ludata
 - PythiaWrapper5_720.h, 330
- ludatr
 - PythiaWrapper5_720.h, 330
- ludatr_
 - PythiaWrapper5_720.h, 332
- luhepc
 - PythiaWrapper5_720.h, 330
- lujets
 - PythiaWrapper5_720.h, 330
- lujets_
 - PythiaWrapper5_720.h, 332
- lulist
 - PythiaWrapper5_720.h, 330
- LWEVT
 - HerwigWrapper6_4.h, 295
- LWSUD
 - HerwigWrapper6_4.h, 295
- m
 - HepMC::FourVector, 53
 - HepMC::HEPEVT_Wrapper, 146
- m2
 - HepMC::FourVector, 53
- m_map_iterator
 - HepMC::GenEvent::particle_—
const_iterator, 88
 - HepMC::GenEvent::particle_—
iterator, 91

- HepMC::GenEvent::vertex_const_iterator, 94
- HepMC::GenEvent::vertex_iterator, 97
- main
 - example_BuildEventFromScratch.cc, 254
 - example_EventSelection.cc, 255
 - example_MyHerwig.cc, 256
 - example_MyPythia.cc, 257
 - example_MyPythiaOnlyToHepMC.cc, 260
 - example_PythiaStreamIO.cc, 261
 - example_UsingIterators.cc, 263
 - testFlow.cc, 361
 - testHerwigCopies.cc, 365
 - testPrintBug.cc, 366
 - testPythiaCopies.cc, 367
 - testSimpleVector.cc, 368
 - testUnits.cc, 369
- make_antiparticles_from_particles
 - HepMC::ParticleDataTable, 215
- mass
 - HepMC::ParticleData, 208
- max_number_entries
 - HepMC::HEPEVT_Wrapper, 146
- MAXER
 - HerwigWrapper6_4.h, 296
- MAXEV
 - HerwigWrapper6_4.h, 296
- MAXPR
 - HerwigWrapper6_4.h, 296
- mdcy
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- mdme
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- merge_table
 - HepMC::ParticleDataTable, 215
- MEV
 - HepMC::Units, 35
- mint
 - PythiaWrapper5_720.h, 332
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- MissingEndKey
 - HepMC::IO_Exception, 163
- MissingStartKey
 - HepMC::IO_Exception, 163
- MM
 - HepMC::Units, 35
- model_independent_pdg_id
 - HepMC::ParticleData, 208
- MODPDF
 - HerwigWrapper6_4.h, 296
- momentum
 - HepMC::GenParticle, 104
- momentum_unit
 - HepMC::GenEvent, 74
- MomentumUnit
 - HepMC::Units, 35
- mpi
 - HepMC::GenEvent, 74
- mrlu
 - PythiaWrapper5_720.h, 332
- mrpy
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- msel
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- mselpd
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 348
- msti
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 349
- mstj
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 349
- mstp
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 349
- mstu
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 340
 - PythiaWrapper6_2.h, 349
- msub
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- n
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- N_Nwounded_collisions
 - HepMC::HeavyIon, 136
- name
 - HepMC::ParticleData, 208
 - HepMC::Units, 36

- NBTRY
 - HerwigWrapper6_4.h, 296
- Ncoll
 - HepMC::HeavyIon, 136
- Ncoll_hard
 - HepMC::HeavyIon, 136
- NCOLO
 - HerwigWrapper6_4.h, 296
- NCTRY
 - HerwigWrapper6_4.h, 296
- NDTRY
 - HerwigWrapper6_4.h, 296
- NETRY
 - HerwigWrapper6_4.h, 296
- NFLAV
 - HerwigWrapper6_4.h, 296
- ngen
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- ngenpd
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- NGSPL
 - HerwigWrapper6_4.h, 296
- no_gaps_in_barcodes
 - HepMC::IO_HERWIG, 177
- normal3d
 - HepMC::Polarization, 227
- NOSPAC
 - HerwigWrapper6_4.h, 296
- not_in_vector
 - HepMC, 27
- NOWGT
 - HerwigWrapper6_4.h, 297
- npad
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- Npart_proj
 - HepMC::HeavyIon, 136
- Npart_targ
 - HepMC::HeavyIon, 137
- NPRFMT
 - HerwigWrapper6_4.h, 297
- NRN
 - HerwigWrapper6_4.h, 297
- NSTRU
 - HerwigWrapper6_4.h, 297
- NSTRY
 - HerwigWrapper6_4.h, 297
- NullEvent
 - HepMC::IO_Exception, 163
- number_children
 - HepMC::HEPEVT_Wrapper, 146
- number_entries
 - HepMC::HEPEVT_Wrapper, 146
- number_parents
 - HepMC::HEPEVT_Wrapper, 147
- NUMER
 - HerwigWrapper6_4.h, 297
- NUMERU
 - HerwigWrapper6_4.h, 297
- NWGTS
 - HerwigWrapper6_4.h, 297
- Nwounded_N_collisions
 - HepMC::HeavyIon, 137
- Nwounded_Nwounded_collisions
 - HepMC::HeavyIon, 137
- NZBIN
 - HerwigWrapper6_4.h, 297
- OK
 - HepMC::IO_Exception, 163
- OMHMX
 - HerwigWrapper6_4.h, 297
- operator *
 - HepMC::GenEvent::particle_
const_iterator, 87
 - HepMC::GenEvent::particle_
iterator, 90
 - HepMC::GenEvent::vertex_const_
iterator, 93
 - HepMC::GenEvent::vertex_
iterator, 96
 - HepMC::GenVertex::edge_
iterator, 124
 - HepMC::GenVertex::particle_
iterator, 127
 - HepMC::GenVertex::vertex_
iterator, 131
- operator HepMC::FourVector
 - HepMC::GenParticle, 104
 - HepMC::GenVertex, 115
- operator HepMC::ThreeVector
 - HepMC::GenVertex, 116
- operator particle_const_iterator
 - HepMC::GenEvent::particle_
iterator, 90
- operator vertex_const_iterator
 - HepMC::GenEvent::vertex_
iterator, 96
- operator!=
 - HepMC::Flow, 47
 - HepMC::FourVector, 53
 - HepMC::GenCrossSection, 62
 - HepMC::GenEvent::particle_
const_iterator, 87

- HepMC::GenEvent::particle_
 iterator, 90
- HepMC::GenEvent::vertex_const_
 iterator, 93
- HepMC::GenEvent::vertex_
 iterator, 97
- HepMC::GenParticle, 104
- HepMC::GenVertex, 116
- HepMC::GenVertex::edge_
 iterator, 124
- HepMC::GenVertex::particle_
 iterator, 127
- HepMC::GenVertex::vertex_
 iterator, 131
- HepMC::HeavyIon, 137
- HepMC::ParticleData, 208
- HepMC::PdfInfo, 221
- HepMC::Polarization, 227
- HepMC::ThreeVector, 240
- operator()
 - IsEventGood, 197
 - IsFinalState, 198
 - IsGoodEvent, 199
 - IsGoodEventMyPythia, 200
 - IsPhoton, 201
 - IsStateFinal, 202
 - IsW_Boson, 203
- operator++
 - HepMC::GenEvent::particle_
 const_iterator, 87, 88
 - HepMC::GenEvent::particle_
 iterator, 91
 - HepMC::GenEvent::vertex_const_
 iterator, 93
 - HepMC::GenEvent::vertex_
 iterator, 97
 - HepMC::GenVertex::edge_
 iterator, 125
 - HepMC::GenVertex::particle_
 iterator, 128
 - HepMC::GenVertex::vertex_
 iterator, 131, 132
- operator<<
 - HepMC, 24–30
 - HepMC::Flow, 49
 - HepMC::GenParticle, 108
 - HepMC::GenVertex, 121
 - HepMC::IO_BaseClass, 159
 - HepMC::ParticleData, 210
 - HepMC::Polarization, 228
- operator=
 - HepMC::Flow, 47
 - HepMC::FourVector, 54
 - HepMC::GenCrossSection, 62
 - HepMC::GenEvent, 74
 - HepMC::GenEvent::particle_
 const_iterator, 88
 - HepMC::GenEvent::particle_
 iterator, 91
 - HepMC::GenEvent::vertex_const_
 iterator, 94
 - HepMC::GenEvent::vertex_
 iterator, 97
 - HepMC::GenParticle, 104
 - HepMC::GenVertex, 116
 - HepMC::GenVertex::edge_
 iterator, 125
 - HepMC::GenVertex::particle_
 iterator, 128
 - HepMC::GenVertex::vertex_
 iterator, 132
 - HepMC::HeavyIon, 137
 - HepMC::ParticleDataTable, 216
 - HepMC::PdfInfo, 221
 - HepMC::Polarization, 227
 - HepMC::ThreeVector, 240
 - HepMC::WeightContainer, 247, 248
- operator==
 - HepMC::Flow, 48
 - HepMC::FourVector, 54
 - HepMC::GenCrossSection, 62
 - HepMC::GenEvent::particle_
 const_iterator, 88
 - HepMC::GenEvent::particle_
 iterator, 91
 - HepMC::GenEvent::vertex_const_
 iterator, 94
 - HepMC::GenEvent::vertex_
 iterator, 97
 - HepMC::GenParticle, 104
 - HepMC::GenVertex, 116
 - HepMC::GenVertex::edge_
 iterator, 125
 - HepMC::GenVertex::particle_
 iterator, 128
 - HepMC::GenVertex::vertex_
 iterator, 132
 - HepMC::HeavyIon, 137
 - HepMC::ParticleData, 208
 - HepMC::PdfInfo, 221
 - HepMC::Polarization, 227
 - HepMC::ThreeVector, 240
- operator>>
 - HepMC, 24–27
 - HepMC::IO_BaseClass, 160
- operator[]
 - HepMC::ParticleDataTable, 216

- HepMC::WeightContainer, 248
- order_begin
 - HepMC::TempParticleMap, 236
- order_end
 - HepMC::TempParticleMap, 236
- orderIterator
 - HepMC::TempParticleMap, 235
- output
 - HepMC::detail, 33, 34
- p
 - PythiaWrapper5_720.h, 333
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- parent_event
 - HepMC::GenParticle, 104
 - HepMC::GenVertex, 116
- parents
 - HepMC, 22
- parf
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- pari
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 349
- parj
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 350
- parp
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 350
- PART1
 - HerwigWrapper6_4.h, 297
- PART2
 - HerwigWrapper6_4.h, 297
- particle_const_iterator
 - HepMC::GenEvent, 84
 - HepMC::GenEvent::particle_
 - const_iterator, 87
- particle_iterator
 - HepMC::GenEvent, 84
 - HepMC::GenEvent::particle_
 - iterator, 90
 - HepMC::GenVertex, 121
 - HepMC::GenVertex::particle_
 - iterator, 127
- particle_owner
 - HepMC::Flow, 48
- ParticleData
 - HepMC::ParticleData, 206
- ParticleData.cc, 319
- ParticleData.h, 320
- ParticleDataTable
 - HepMC::ParticleDataTable, 213
- ParticleDataTable.h, 321
- particles_begin
 - HepMC::GenEvent, 74
 - HepMC::GenVertex, 117
- particles_empty
 - HepMC::GenEvent, 75
- particles_end
 - HepMC::GenEvent, 75
 - HepMC::GenVertex, 117
- particles_in_const_begin
 - HepMC::GenVertex, 117
- particles_in_const_end
 - HepMC::GenVertex, 117
- particles_in_const_iterator
 - HepMC::GenVertex, 112
- particles_in_size
 - HepMC::GenVertex, 117
- particles_out_const_begin
 - HepMC::GenVertex, 117
- particles_out_const_end
 - HepMC::GenVertex, 118
- particles_out_const_iterator
 - HepMC::GenVertex, 112
- particles_out_size
 - HepMC::GenVertex, 118
- particles_size
 - HepMC::GenEvent, 75
- particleTypes
 - testHepMCMethods.cc, 363
 - testHepMCMethods.h, 364
- paru
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 350
- PBEAM1
 - HerwigWrapper6_4.h, 298
- PBEAM2
 - HerwigWrapper6_4.h, 298
- pdf1
 - HepMC::PdfInfo, 221
- pdf2
 - HepMC::PdfInfo, 221
- pdf_id1
 - HepMC::PdfInfo, 221
- pdf_id2
 - HepMC::PdfInfo, 222
- pdf_info
 - HepMC::GenEvent, 75, 76
- PdfInfo
 - HepMC::PdfInfo, 220

- PdfInfo.cc, 322
- PdfInfo.h, 323
- pdg_id
 - HepMC::GenParticle, 105
 - HepMC::ParticleData, 209
- PDIQK
 - HerwigWrapper6_4.h, 298
- perp
 - HepMC::FourVector, 54
 - HepMC::ThreeVector, 240
- perp2
 - HepMC::FourVector, 54
 - HepMC::ThreeVector, 240
- PGSMX
 - HerwigWrapper6_4.h, 298
- PGSPL
 - HerwigWrapper6_4.h, 298
- PH3MIX
 - HerwigWrapper6_4.h, 298
- phi
 - HepMC::FourVector, 54
 - HepMC::Polarization, 227
 - HepMC::ThreeVector, 240
- PHIMIX
 - HerwigWrapper6_4.h, 298
- PIFAC
 - HerwigWrapper6_4.h, 298
- pmas
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 341
 - PythiaWrapper6_2.h, 350
- point3d
 - HepMC::GenVertex, 118
- Polarization
 - HepMC::Polarization, 226
- polarization
 - HepMC::GenParticle, 105
- Polarization.cc, 324
- Polarization.h, 325
- pop_back
 - HepMC::WeightContainer, 248
- position
 - HepMC::GenVertex, 118
- precision
 - HepMC::IO_GenEvent, 167
- print
 - HepMC::Flow, 48
 - HepMC::GenEvent, 76
 - HepMC::GenParticle, 105
 - HepMC::GenVertex, 118
 - HepMC::IO_AsciiParticles, 155
 - HepMC::IO_BaseClass, 160
 - HepMC::IO_GenEvent, 167
 - HepMC::IO_HEPEVT, 172
 - HepMC::IO_HERWIG, 177
 - HepMC::IO_PDG_ParticleData-Table, 182
 - HepMC::ParticleData, 209
 - HepMC::ParticleDataTable, 216
 - HepMC::Polarization, 227
 - HepMC::WeightContainer, 248
- print_hepevt
 - HepMC::HEPEVT_Wrapper, 147
- print_hepevt_particle
 - HepMC::HEPEVT_Wrapper, 147
- print_inconsistency_errors
 - HepMC::IO_HEPEVT, 172
 - HepMC::IO_HERWIG, 177
- print_legend
 - HepMC::HEPEVT_Wrapper, 147
- print_version
 - HepMC::GenEvent, 76
- PRNDEC
 - HerwigWrapper6_4.h, 298
- PRNDEF
 - HerwigWrapper6_4.h, 298
- PRNTEX
 - HerwigWrapper6_4.h, 298
- PRNWEB
 - HerwigWrapper6_4.h, 298
- production_vertex
 - HepMC::GenParticle, 105
- PRSOF
 - HerwigWrapper6_4.h, 299
- PRVTX
 - HerwigWrapper6_4.h, 299
- pseudoRapidity
 - HepMC::FourVector, 54
- PSPLT
 - HerwigWrapper6_4.h, 299
- PTRMS
 - HerwigWrapper6_4.h, 299
- push_back
 - HepMC::WeightContainer, 248
- px
 - HepMC::FourVector, 54
 - HepMC::HEPEVT_Wrapper, 147
- PXRMS
 - HerwigWrapper6_4.h, 299
- py
 - HepMC::FourVector, 55
 - HepMC::HEPEVT_Wrapper, 148
- pydat1
 - PythiaWrapper6_152.h, 337
 - PythiaWrapper6_2.h, 346
- pydat1_
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350

- pydat2
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pydat2_
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pydat3
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pydat3_
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pydata
 - PythiaWrapper5_720.h, 330
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pydatr
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pydatr_
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pyevnt
 - PythiaWrapper5_720.h, 330
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyhepc
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyinit
 - PythiaWrapper5_720.h, 330
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyint1
 - PythiaWrapper5_720.h, 330
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyint1_
 - PythiaWrapper5_720.h, 334
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pyint2
 - PythiaWrapper5_720.h, 330
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyint2_
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pyint5
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyint5_
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pyjets
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 346
- pyjets_
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pyjets_maxn
 - PythiaWrapper6_2.h, 350
- pylist
 - PythiaWrapper6_152.h, 338
 - PythiaWrapper6_2.h, 347
- pypars
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- pypars_
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pystat
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- pysubs
 - PythiaWrapper5_720.h, 331
 - PythiaWrapper6_152.h, 339
 - PythiaWrapper6_2.h, 347
- pysubs_
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- pythia_in
 - example_MyPythia.cc, 257
- pythia_in_out
 - example_MyPythia.cc, 258
- pythia_out
 - example_MyPythia.cc, 258
- pythia_particle_out
 - example_MyPythia.cc, 258
- PythiaHelper.h, 326
- PythiaHelper.h
 - getPythiaCrossSection, 326
 - initPythia, 326
- PythiaWrapper.h, 327
- PythiaWrapper5_720.h, 328
- PythiaWrapper5_720.h
 - brat, 331
 - ckin, 331
 - coef, 331
 - icol, 331
 - initpydata, 329

iset, 331
k, 332
kchg, 332
kfdp, 332
kfin, 332
kfpr, 332
ludat1, 329
ludat1_, 332
ludat2, 330
ludat2_, 332
ludat3, 330
ludat3_, 332
ludata, 330
ludatr, 330
ludatr_, 332
luhepc, 330
lujets, 330
lujets_, 332
lulist, 330
mdcy, 332
mdme, 332
mint, 332
mrlu, 332
msel, 333
msti, 333
mstj, 333
mstp, 333
mstu, 333
msub, 333
n, 333
ngen, 333
p, 333
parf, 334
pari, 334
parj, 334
parp, 334
paru, 334
pmas, 334
pydata, 330
pyevnt, 330
pyinit, 330
pyint1, 330
pyint1_, 334
pyint2, 330
pyint2_, 335
pyint5, 331
pyint5_, 335
pypars, 331
pypars_, 335
pystat, 331
pysubs, 331
pysubs_, 335
rrlu, 335
v, 335
vckm, 335
vint, 335
xsec, 335
PythiaWrapper6_152.h, 336
PythiaWrapper6_152.h
brat, 339
ckin, 339
coef, 339
icol, 339
initpydata, 337
iset, 339
k, 339
kchg, 339
kfdp, 339
kfin, 340
kfpr, 340
mdcy, 340
mdme, 340
mint, 340
mrpy, 340
msel, 340
mselpd, 340
msti, 340
mstj, 340
mstp, 340
mstu, 340
msub, 341
n, 341
ngen, 341
ngenpd, 341
npad, 341
p, 341
parf, 341
pari, 341
parj, 341
parp, 341
paru, 341
pmas, 341
pydat1, 337
pydat1_, 342
pydat2, 338
pydat2_, 342
pydat3, 338
pydat3_, 342
pydata, 338
pydatr, 338
pydatr_, 342
pyevnt, 338
pyhepc, 338
pyinit, 338
pyint1, 338
pyint1_, 342
pyint2, 338
pyint2_, 342

- pyint5, 338
- pyint5_, 342
- pyjets, 338
- pyjets_, 342
- pylist, 338
- pypars, 339
- pypars_, 342
- pystat, 339
- pysubs, 339
- pysubs_, 342
- rrpy, 342
- v, 342
- vckm, 342
- vint, 342
- xsec, 342
- PythiaWrapper6_152_WIN32.h, 343
- PythiaWrapper6_2.h, 344
- PythiaWrapper6_2.h
 - brat, 347
 - ckin, 347
 - coef, 347
 - icol, 347
 - initpydata, 346, 347
 - iset, 348
 - k, 348
 - kchg, 348
 - kfdp, 348
 - kfin, 348
 - kfpr, 348
 - mdcy, 348
 - mdme, 348
 - mint, 348
 - mrpy, 348
 - msei, 348
 - mselpd, 348
 - msti, 349
 - mstj, 349
 - mstp, 349
 - mstu, 349
 - msub, 349
 - n, 349
 - ngen, 349
 - ngenpd, 349
 - npad, 349
 - p, 349
 - parf, 349
 - pari, 349
 - parj, 350
 - parp, 350
 - paru, 350
 - pmas, 350
 - pydat1, 346
 - pydat1_, 350
 - pydat2, 346
 - pydat2_, 350
 - pydat3, 346
 - pydat3_, 350
 - pydata, 346
 - pydatr, 346
 - pydatr_, 350
 - pyevnt, 346
 - pyhepc, 346
 - pyinit, 346
 - pyint1, 346
 - pyint1_, 350
 - pyint2, 346
 - pyint2_, 350
 - pyint5, 346
 - pyint5_, 350
 - pyjets, 346
 - pyjets_, 350
 - pyjets_maxn, 350
 - pylist, 347
 - pypars, 347
 - pypars_, 350
 - pystat, 347
 - pysubs, 347
 - pysubs_, 350
 - rrpy, 350
 - uvevnt, 347
 - upinit, 347
 - v, 350
 - vckm, 351
 - vint, 351
 - xsec, 351
- PythiaWrapper6_2_WIN32.h, 352
- pz
 - HepMC::FourVector, 55
 - HepMC::HEPEVT_Wrapper, 148
- QCDL3
 - HerwigWrapper6_4.h, 299
- QCDL5
 - HerwigWrapper6_4.h, 299
- QCDSLAM
 - HerwigWrapper6_4.h, 299
- QDIQK
 - HerwigWrapper6_4.h, 299
- QFCH
 - HerwigWrapper6_4.h, 299
- QG
 - HerwigWrapper6_4.h, 299
- QSPAC
 - HerwigWrapper6_4.h, 299
- QV
 - HerwigWrapper6_4.h, 300
- r

HepMC::ThreeVector, 241
 random_states
 HepMC::GenEvent, 76
 range
 HepMC::GenVertex::vertex_
 iterator, 132
 rdstate
 HepMC::IO_AsciiParticles, 156
 HepMC::IO_GenEvent, 167
 HepMC::IO_PDG_ParticleData-
 Table, 182
 read
 HepMC::GenCrossSection, 62
 HepMC::GenEvent, 76
 read_entry
 HepMC::IO_PDG_ParticleData-
 Table, 182
 read_io_particle_data_table
 HepMC::IO_GenEvent, 167
 read_next_event
 HepMC::IO_BaseClass, 160
 read_particle
 HepMC::detail, 33
 read_particle_data
 HepMC::IO_GenEvent, 167
 read_particle_data_table
 HepMC::IO_BaseClass, 160
 read_units
 HepMC::detail, 34
 read_vertex
 HepMC::detail, 33
 readPythiaStreamIO
 example_PythiaStreamIO.cc, 261
 relatives
 HepMC, 23
 remove_barcode
 HepMC::GenEvent, 77
 remove_gaps_in_hepevt
 HepMC::IO_HERWIG, 177
 remove_particle
 HepMC::GenVertex, 119
 remove_particle_in
 HepMC::GenVertex, 119
 remove_particle_out
 HepMC::GenVertex, 119
 remove_vertex
 HepMC::GenEvent, 77
 repair_hepevt
 HepMC::IO_HERWIG, 178
 rho
 HepMC::FourVector, 55
 rrlu
 PythiaWrapper5_720.h, 335
 rrpv
 PythiaWrapper6_152.h, 342
 PythiaWrapper6_2.h, 350
 SCABI
 HerwigWrapper6_4.h, 300
 scalePDF
 HepMC::PdfInfo, 222
 search_for_key_end
 HepMC::IO_PDG_ParticleData-
 Table, 183
 SearchVector.cc, 353
 SearchVector.h, 354
 set
 HepMC::FourVector, 55
 HepMC::ThreeVector, 241
 set_alphaQCD
 HepMC::GenEvent, 77
 set_alphaQED
 HepMC::GenEvent, 78
 set_barcode
 HepMC::GenEvent, 78
 set_barcode_
 HepMC::GenParticle, 105
 HepMC::GenVertex, 119
 set_beam_particles
 HepMC::GenEvent, 78
 set_charge
 HepMC::ParticleData, 209
 set_children
 HepMC::HEPEVT_Wrapper, 148
 set_clifetime
 HepMC::ParticleData, 209
 set_cross_section
 HepMC::GenCrossSection, 62
 HepMC::GenEvent, 78
 set_cross_section_error
 HepMC::GenCrossSection, 63
 set_description
 HepMC::ParticleDataTable, 216
 set_eccentricity
 HepMC::HeavyIon, 138
 set_end_vertex_
 HepMC::GenParticle, 105
 set_event_number
 HepMC::GenEvent, 79
 HepMC::HEPEVT_Wrapper, 148
 set_event_plane_angle
 HepMC::HeavyIon, 138
 set_event_scale
 HepMC::GenEvent, 79
 set_finished_first_event
 HepMC::StreamInfo, 233
 set_flow
 HepMC::GenParticle, 106

set_generated_mass	HepMC::GenParticle, 106	HepMC::HEPEVT_Wrapper, 150
set_has_key	HepMC::StreamInfo, 233	set_pdf1
set_heavy_ion	HepMC::GenEvent, 79	HepMC::PdfInfo, 222
set_icode	HepMC::Flow, 48	set_pdf2
set_id	HepMC::GenVertex, 119	HepMC::PdfInfo, 222
	HepMC::HEPEVT_Wrapper, 148	set_pdf_id1
set_id1	HepMC::PdfInfo, 222	HepMC::PdfInfo, 222
set_id2	HepMC::PdfInfo, 222	set_pdf_id2
set_impact_parameter	HepMC::HeavyIon, 138	HepMC::PdfInfo, 223
set_input_units	HepMC, 25	set_pdf_info
set_io_type	HepMC::StreamInfo, 234	HepMC::GenEvent, 80
set_mass	HepMC::HEPEVT_Wrapper, 149	set_pdg_id
	HepMC::ParticleData, 209	HepMC::GenParticle, 106
set_max_number_entries	HepMC::HEPEVT_Wrapper, 149	set_phi
set_momentum	HepMC::GenParticle, 106	HepMC::Polarization, 227
	HepMC::HEPEVT_Wrapper, 149	set_polarization
set_mpi	HepMC::GenEvent, 79	HepMC::GenParticle, 106
set_N_Nwounded_collisions	HepMC::HeavyIon, 138	set_position
set_Ncoll	HepMC::HeavyIon, 138	HepMC::GenVertex, 120
set_Ncoll_hard	HepMC::HeavyIon, 138	HepMC::HEPEVT_Wrapper, 150
set_no_gaps_in_barcodes	HepMC::IO_HERWIG, 178	set_print_inconsistency_errors
set_normal3d	HepMC::Polarization, 227	HepMC::IO_HEPEVT, 172
set_Npart_proj	HepMC::HeavyIon, 138	HepMC::IO_HERWIG, 179
set_Npart_targ	HepMC::HeavyIon, 139	set_production_vertex
set_number_entries	HepMC::HEPEVT_Wrapper, 149	HepMC::GenParticle, 107
set_Nwounded_N_collisions	HepMC::HeavyIon, 139	set_random_states
set_Nwounded_Nwounded_collisions	HepMC::HeavyIon, 139	HepMC::GenEvent, 80
set_parent_event	HepMC::GenVertex, 120	set_scalePDF
set_parents		HepMC::PdfInfo, 223
		set_sigma_inel_NN
		HepMC::HeavyIon, 139
		set_signal_process_id
		HepMC::GenEvent, 80
		set_signal_process_vertex
		HepMC::GenEvent, 80
		set_sizeof_int
		HepMC::HEPEVT_Wrapper, 150
		set_sizeof_real
		HepMC::HEPEVT_Wrapper, 150
		set_spectator_neutrons
		HepMC::HeavyIon, 139
		set_spectator_protons
		HepMC::HeavyIon, 139
		set_spin
		HepMC::ParticleData, 209
		set_status
		HepMC::GenParticle, 107
		HepMC::HEPEVT_Wrapper, 150
		set_theta
		HepMC::Polarization, 228
		set_theta_phi
		HepMC::Polarization, 228
		set_trust_beam_particles
		HepMC::IO_HEPEVT, 172

set_trust_both_mothers_and_daughters
 HepMC::IO_HEPEVT, 172
 HepMC::IO_HERWIG, 179
 set_trust_mothers_before_daughters
 HepMC::IO_HEPEVT, 173
 HepMC::IO_HERWIG, 179
 set_unique_icode
 HepMC::Flow, 48
 set_width
 HepMC::ParticleData, 209
 set_x1
 HepMC::PdfInfo, 223
 set_x2
 HepMC::PdfInfo, 223
 setE
 HepMC::FourVector, 56
 setGeneratedMass
 HepMC::GenParticle, 107
 setPhi
 HepMC::ThreeVector, 241
 setPrecision
 HepMC::IO_AsciiParticles, 156
 setPx
 HepMC::FourVector, 56
 setPy
 HepMC::FourVector, 56
 setPz
 HepMC::FourVector, 56
 setT
 HepMC::FourVector, 56
 setTheta
 HepMC::ThreeVector, 241
 setX
 HepMC::FourVector, 57
 HepMC::ThreeVector, 241
 setY
 HepMC::FourVector, 57
 HepMC::ThreeVector, 242
 setZ
 HepMC::FourVector, 57
 HepMC::ThreeVector, 242
 sigma_inel_NN
 HepMC::HeavyIon, 139
 signal_process_id
 HepMC::GenEvent, 80
 signal_process_vertex
 HepMC::GenEvent, 81
 SimpleVector.h, 355
 size
 HepMC::Flow, 48
 HepMC::ParticleDataTable, 217
 HepMC::WeightContainer, 248
 sizeof_int
 HepMC::HEPEVT_Wrapper, 151
 sizeof_real
 HepMC::HEPEVT_Wrapper, 151
 SOFTME
 HerwigWrapper6_4.h, 300
 spectator_neutrons
 HepMC::HeavyIon, 140
 spectator_protons
 HepMC::HeavyIon, 140
 spin
 HepMC::ParticleData, 210
 status
 HepMC::GenParticle, 107
 HepMC::HEPEVT_Wrapper, 151
 stream_id
 HepMC::StreamInfo, 234
 StreamHelpers.cc, 356
 StreamHelpers.h, 357
 StreamInfo
 HepMC::StreamInfo, 231
 StreamInfo.cc, 358
 StreamInfo.h, 359
 suggest_barcode
 HepMC::GenParticle, 107
 HepMC::GenVertex, 120
 swap
 HepMC::Flow, 48
 HepMC::FourVector, 57
 HepMC::GenCrossSection, 63
 HepMC::GenEvent, 81
 HepMC::GenParticle, 107
 HepMC::GenVertex, 120
 HepMC::HeavyIon, 140
 HepMC::PdfInfo, 223
 HepMC::Polarization, 228
 HepMC::ThreeVector, 242
 HepMC::WeightContainer, 248
 SWEIN
 HerwigWrapper6_4.h, 300
 t
 HepMC::FourVector, 57
 HepMC::HEPEVT_Wrapper, 151
 TempMap
 HepMC::TempParticleMap, 235
 TempMapIterator
 HepMC::TempParticleMap, 235
 TempOrderMap
 HepMC::TempParticleMap, 235
 TempParticleMap
 HepMC::TempParticleMap, 236
 TempParticleMap.h, 360
 testFlow.cc, 361
 testFlow.cc

- FlowVec, 361
- main, 361
- testHepMCIteration.h, 362
- testHepMCIteration.h
 - IsPhoton, 362
 - IsWBoson, 362
- testHepMCMethods.cc, 363
- testHepMCMethods.cc
 - findPiZero, 363
 - particleTypes, 363
- testHepMCMethods.h, 364
- testHepMCMethods.h
 - findPiZero, 364
 - particleTypes, 364
- testHerwigCopies.cc, 365
- testHerwigCopies.cc
 - main, 365
- testPrintBug.cc, 366
- testPrintBug.cc
 - main, 366
- testPythiaCopies.cc, 367
- testPythiaCopies.cc
 - main, 367
- testSimpleVector.cc, 368
- testSimpleVector.cc
 - main, 368
- testUnits.cc, 369
- testUnits.cc
 - main, 369
- theta
 - HepMC::FourVector, 58
 - HepMC::Polarization, 228
 - HepMC::ThreeVector, 242
- ThreeVector
 - HepMC::ThreeVector, 239, 240
- TLOUT
 - HerwigWrapper6_4.h, 300
- TMTOP
 - HerwigWrapper6_4.h, 300
- translate_herwig_to_pdg_id
 - HepMC::IO_HERWIG, 179
- trust_beam_particles
 - HepMC::IO_HEPEVT, 173
- trust_both_mothers_and_daughters
 - HepMC::IO_HEPEVT, 173
 - HepMC::IO_HERWIG, 179
- trust_mothers_before_daughters
 - HepMC::IO_HEPEVT, 173
 - HepMC::IO_HERWIG, 179
- type
 - HepMC::detail::disable_if<false, T>, 40
 - HepMC::detail::enable_if<true, T>, 42
- Units, 37
- Units.h, 370
- upevnt
 - PythiaWrapper6_2.h, 347
- upinit
 - PythiaWrapper6_2.h, 347
- use_input_units
 - HepMC::IO_GenEvent, 167
 - HepMC::StreamInfo, 234
- use_units
 - HepMC::GenEvent, 81
- v
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 350
- valid_beam_particles
 - HepMC::GenEvent, 81
- value
 - HepMC::detail::is_arithmetic, 184
 - HepMC::detail::is_arithmetic<char>, 185
 - HepMC::detail::is_arithmetic<double>, 186
 - HepMC::detail::is_arithmetic<float>, 187
 - HepMC::detail::is_arithmetic<int>, 188
 - HepMC::detail::is_arithmetic<long>, 189
 - HepMC::detail::is_arithmetic<long double>, 190
 - HepMC::detail::is_arithmetic<short>, 191
 - HepMC::detail::is_arithmetic<signed char>, 192
 - HepMC::detail::is_arithmetic<unsigned char>, 193
 - HepMC::detail::is_arithmetic<unsigned int>, 194
 - HepMC::detail::is_arithmetic<unsigned long>, 195
 - HepMC::detail::is_arithmetic<unsigned short>, 196
- VCKM
 - HerwigWrapper6_4.h, 300
- vckm
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 351
- VectorConversion.h, 371
- VectorConversion.h
 - convertTo, 371

- version
 - HepMC, 28
- Version.h, 372
- versionName
 - HepMC, 28
- vertex_const_iterator
 - HepMC::GenEvent, 84
 - HepMC::GenEvent::vertex_const_iterator, 93
- vertex_iterator
 - HepMC::GenEvent, 85
 - HepMC::GenEvent::vertex_iterator, 96
 - HepMC::GenVertex, 122
 - HepMC::GenVertex::vertex_iterator, 130
- vertex_root
 - HepMC::GenVertex::edge_iterator, 125
 - HepMC::GenVertex::vertex_iterator, 132
- vertices_begin
 - HepMC::GenEvent, 82
 - HepMC::GenVertex, 121
- vertices_empty
 - HepMC::GenEvent, 82
- vertices_end
 - HepMC::GenEvent, 82
 - HepMC::GenVertex, 121
- vertices_size
 - HepMC::GenEvent, 83
- VFCH
 - HerwigWrapper6_4.h, 300
- VGCUT
 - HerwigWrapper6_4.h, 300
- vint
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 351
- VPCUT
 - HerwigWrapper6_4.h, 300
- VQCUT
 - HerwigWrapper6_4.h, 300
- WBGST
 - HerwigWrapper6_4.h, 300
- WeightContainer
 - HepMC::WeightContainer, 246
- WeightContainer.h, 373
- weights
 - HepMC::GenEvent, 83
 - HepMC::GenVertex, 121
- WGTMAX
 - HerwigWrapper6_4.h, 301
- WGTSUM
 - HerwigWrapper6_4.h, 301
- width
 - HepMC::ParticleData, 210
- write
 - HepMC::GenCrossSection, 63
 - HepMC::GenEvent, 83
- write_byte_num
 - HepMC::HEPEVT_Wrapper, 151
- write_comment
 - HepMC::IO_AsciiParticles, 156
 - HepMC::IO_GenEvent, 168
- write_cross_section
 - HepMC::GenEvent, 83
- write_end_listing
 - HepMC::IO_AsciiParticles, 156
- write_event
 - HepMC::IO_AsciiParticles, 156
 - HepMC::IO_BaseClass, 161
 - HepMC::IO_GenEvent, 168
 - HepMC::IO_HEPEVT, 173
- write_HepMC_IO_block_begin
 - HepMC, 25
- write_HepMC_IO_block_end
 - HepMC, 26
- write_particle_data_table
 - HepMC::IO_AsciiParticles, 156
 - HepMC::IO_BaseClass, 161
 - HepMC::IO_GenEvent, 168
- write_units
 - HepMC::GenEvent, 84
- writePythiaStreamIO
 - example_PythiaStreamIO.cc, 261
- writeVersion
 - HepMC, 28
- WrongFileType
 - HepMC::IO_Exception, 163
- WSQSUM
 - HerwigWrapper6_4.h, 301
- x
 - HepMC::FourVector, 58
 - HepMC::HEPEVT_Wrapper, 152
 - HepMC::ThreeVector, 242
- x1
 - HepMC::PdfInfo, 223
- x2
 - HepMC::PdfInfo, 223
- xsec
 - PythiaWrapper5_720.h, 335
 - PythiaWrapper6_152.h, 342
 - PythiaWrapper6_2.h, 351
- y

HepMC::FourVector, 58
HepMC::HEPEVT_Wrapper, 152
HepMC::ThreeVector, 243

z

HepMC::FourVector, 58
HepMC::HEPEVT_Wrapper, 152
HepMC::ThreeVector, 243

ZBINM

HerwigWrapper6_4.h, 301

zero_everything

HepMC::HEPEVT_Wrapper, 152

zero_hepevt_entry

HepMC::IO_HERWIG, 179

ZPRIME

HerwigWrapper6_4.h, 301