HepMC Reference Manual

2.01.08

Generated by Doxygen 1.5.1-3

Tue Oct 23 17:05:34 2007

Contents

Chapter 1

HepMC Directory Hierarchy

1.1 HepMC Directories

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

Chapter 2

HepMC Namespace Index

2.1 HepMC Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

HepMC Hierarchical Index

3.1 HepMC Class Hierarchy

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

Chapter 4

HepMC Class Index

4.1 HepMC Class List

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

Chapter 5

HepMC File Index

5.1 HepMC File List

Here is a list of all files with brief descriptions:

Chapter 6

HepMC Directory Documentation

6.1 /home/cepa01/garren/lcg/hepmc/HepMC-2.01.08/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_MyPythiaRead.cc**
- file **example_MyPythiaWithEventSelection.cc**
- file **example_PythiaParticle.cc**
- file **example_ReadPDGtable.cc**
- file **example_UsingIterators.cc**
- file **initPythia.cc**
- file **list_of_examples.cc**
- file **PythiaHelper.h**
- file **VectorConversion.h**

6.2 /home/cepa01/garren/lcg/hepmc/HepMC-2.01.08/fio/ Directory Reference

Files

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

6.3 /home/cepa01/garren/lcg/hepmc/HepMC-2.01.08/HepMC/ Directory Reference

Files

- file **enable_if.h**
- file **Flow.h**
- file **GenEvent.h**
- file **GenParticle.h**
- file **GenVertex.h**
- file **HeavyIon.h**
- file **HEPEVT_Wrapper.h**
- file **HepMC_CLHEP20.h**
- file **HerwigWrapper.h**
- file **HerwigWrapper6_4.h**
- file **IO_Ascii.h**
- file **IO_AsciiParticles.h**
- file **IO_BaseClass.h**
- file **IO_ExtendedAscii.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 **TempParticleMap.h**
- file **Version.h**
- file **WeightContainer.h**

6.4 /home/cepa01/garren/lcg/hepmc/HepMC-2.01.08/src/ Directory Reference

Files

- file **Flow.cc**
- file **GenEvent.cc**
- file **GenParticle.cc**
- file **GenVertex.cc**
- file **IO_Ascii.cc**
- file **IO_AsciiParticles.cc**
- file **IO_ExtendedAscii.cc**
- file **IO_PDG_ParticleDataTable.cc**
- file **ParticleData.cc**
- file **Polarization.cc**
- file **SearchVector.cc**

6.5 /home/cepa01/garren/lcg/hepmc/HepMC-2.01.08/test/ Directory Reference

Files

- file **IsGoodEvent.h**
- file **testHepMCIteration.h**
- file **testPrintBug.cc**
- file **testSimpleVector.cc**

Chapter 7

HepMC Namespace Documentation

7.1 CLHEP Namespace Reference

7.2 detail Namespace Reference

7.2.1 Detailed Description

internal namespace

7.3 HepMC Namespace Reference

Classes

- class **Flow**
  The flow object.
- class **GenEvent**
  *The **GenEvent** (p. 46) class is the core of **HepMC** (p. 19).*
- class **GenParticle**
  *The **GenParticle** (p. 77) class contains information about generated particles.*
- class **GenVertex**
  ***GenVertex** (p. 87) contains information about decay vertices.*
- class **HeavyIon**
  *The **HeavyIon** (p. 112) class stores information about heavy ions.*
- class **HEPEVT_Wrapper**
  Generic Wrapper for the fortran HEPEVT common block.
- class **IO_Ascii**
  ***IO_Ascii** (p. 133) is used to read or write from an ascii file.*
- 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. 144)*
- class **IO_ExtendedAscii**
  ***IO_ExtendedAscii** (p. 148) also deals with **HeavyIon** (p. 112) and **PdfInfo** (p. 204).*
- class **IO_HEPEVT**
  HEPEVT IO class.
- class **IO_HERWIG**
  ***IO_HERWIG** (p. 162) is used to get Herwig information.*
- class **IO_PDG_ParticleDataTable**
  *an example **ParticleDataTable** (p. 197) IO method*
- class **ParticleData**
  *an example **ParticleData** (p. 190) class*
- class **ParticleDataTable**
  *an example **ParticleDataTable** (p. 197) class*
- class **PdfInfo**

*The **PdfInfo** (p. 204) class stores PDF information.*

- class **Polarization**

  *The **Polarization** (p. 209) class stores theta and phi for a **GenParticle** (p. 77).*

- class **FourVector**

  ***FourVector** (p. 38) is a simple representation of a physics 4 vector.*

- class **ThreeVector**

  ***ThreeVector** (p. 216) is a simple representation of a position or displacement 3 vector.*

- class **TempParticleMap**

  ***TempParticleMap** (p. 213) is a temporary GenParticle* container used during input.*

- class **WeightContainer**

  Container for the Weights associated with an event or vertex.

Namespaces

- namespace **detail**

Enumerations

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

  type of iteration

Functions

- 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

- double **clifetime_from_width** (double width)

  set lifetime from width

- 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< **HepMC::GenParticle** * > *, **GenParticle** *)
- 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
- 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
- bool **not_in_vector** (std::vector< **GenParticle** * > *v, **GenParticle** *p)
- std::vector< **HepMC::GenParticle** * >::iterator **already_in_vector** (std::vector< **GenParticle** * > *v, **GenParticle** *p)
  *returns true if **GenParticle** (p. 77) is in the vector*

Variables

- static const double **HepMC_hbarc**
  *hbar * c -> 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.

7.3.3 Function Documentation

7.3.3.1 std::vector<HepMC::GenParticle*>::iterator HepMC::already_in_vector (std::vector< GenParticle * > * *v*, GenParticle * *p*)

returns true if **GenParticle** (p. 77) is in the vector

Definition at line 18 of file SearchVector.cc.

References p.

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

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

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

7.3.3.3 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().

7.3.3.4 template<class InputIterator, class OutputIterator, class Predicate> void HepMC::copy_if (InputIterator *first*, InputIterator *last*, OutputIterator *out*, Predicate *pred*) [inline]

define the type of iterator to use

Examples:

example_UsingIterators.cc.

Definition at line 50 of file GenEvent.h.

Referenced by main().

7.3.3.5 bool HepMC::not_in_vector (std::vector< GenParticle * > * *v*, GenParticle * *p*)

Definition at line 11 of file SearchVector.cc.

References already_in_vector(), and p.

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

returns true if it cannot find GenParticle* in the vector

7.3.3.7 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().

7.3.3.8 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().

7.3.3.9 std::ostream& HepMC::operator<< (std::ostream & *ostr*, const GenVertex & *vtx*)

print vertex information

Definition at line 429 of file GenVertex.cc.

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

7.3.3.10 std::ostream& HepMC::operator<< (std::ostream & *ostr*, const GenParticle & *part*)

print particle

Definition at line 195 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().

7.3.3.11 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.

7.3.3.12 void HepMC::version () [inline]

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

Definition at line 26 of file Version.h.

References versionName().

7.3.3.13 std::string HepMC::versionName () [inline]

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

Definition at line 21 of file Version.h.

Referenced by version(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_AsciiParticles::write_event(), HepMC::IO_Ascii::write_event(), and writeVersion().

7.3.3.14 void HepMC::writeVersion (std::ostream & *os*) [inline]

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

Definition at line 32 of file Version.h.

References versionName().

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

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

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. 31) 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. 31)*
- 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. 31) patterns are coded with an integer, in the same manner as in Herwig.

Definition at line 67 of file Flow.h.

8.5.2 Member Typedef Documentation

8.5.2.1 typedef std::map<int,int>::iterator HepMC::Flow::iterator

iterator for flow pattern container

Definition at line 127 of file Flow.h.

8.5.2.2 typedef std::map<int,int>::const_iterator HepMC::Flow::const_iterator

const iterator for flow pattern container

Definition at line 129 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 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().

8.5.4.2 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.

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

equality

Definition at line 193 of file Flow.h.

References m_icode.

8.5.4.4 bool HepMC::Flow::operator!= (const Flow & *a*) const [inline]

inequality

Definition at line 199 of file Flow.h.

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

print **Flow** (p. 31) information to ostr

Definition at line 34 of file Flow.cc.

8.5.4.6 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().

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().

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

find particle owning this **Flow** (p. 31)

Definition at line 161 of file Flow.h.

8.5.4.9 int HepMC::Flow::icode (int *code_index* = 1) const [inline]

flow code

Definition at line 164 of file Flow.h.

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

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

set flow code

Definition at line 168 of file Flow.h.

Referenced by HepMC::IO_ExtendedAscii::read_particle(), HepMC::IO_Ascii::read_particle(), and HepMC::GenParticle::set_flow().

8.5.4.11 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 172 of file Flow.h.

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

8.5.4.12 bool HepMC::Flow::empty () const [inline]

return true if there is no flow container

Definition at line 178 of file Flow.h.

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

size of flow pattern container

Definition at line 179 of file Flow.h.

8.5.4.14 void HepMC::Flow::clear () [inline]

clear flow patterns

Definition at line 180 of file Flow.h.

8.5.4.15 bool HepMC::Flow::erase (int *code_index*) [inline]

empty flow pattern container

Definition at line 181 of file Flow.h.

8.5.4.16 Flow::iterator HepMC::Flow::begin () [inline]

beginning of flow pattern container

Definition at line 184 of file Flow.h.

8.5.4.17 Flow::iterator HepMC::Flow::end () [inline]

end of flow pattern container

Definition at line 185 of file Flow.h.

8.5.4.18 Flow::const_iterator HepMC::Flow::begin () const [inline]

beginning of flow pattern container

Definition at line 186 of file Flow.h.

8.5.4.19 Flow::const_iterator HepMC::Flow::end () const [inline]

end of flow pattern container

Definition at line 187 of file Flow.h.

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

for internal use only

8.5.4.21 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

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. 38) 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. 42) 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 **mag** () const

  Magnitude of the spatial vector.

- 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. 38) is a simple representation of a physics 4 vector.

For compatibility with existing code, the basic expected geometrical access methods are povided. Also, there is a templated constructor that will take another vector (HepLorentzVector, GenVector, ...) which must have the following methods: **x()** (p. 41), **y()** (p. 42), **z()** (p. 42), **t()** (p. 42).

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

swap

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

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

return px

Definition at line 70 of file SimpleVector.h.

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

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

return py

Definition at line 71 of file SimpleVector.h.

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

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

return pz

Definition at line 72 of file SimpleVector.h.

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

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

return E

Definition at line 73 of file SimpleVector.h.

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

8.6.3.6 double HepMC::FourVector::x () const [inline]

return x

Definition at line 75 of file SimpleVector.h.

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

8.6.3.7 double HepMC::FourVector::y () const [inline]

return y

Definition at line 76 of file SimpleVector.h.

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

8.6.3.8 double HepMC::FourVector::z () const [inline]

return z

Definition at line 77 of file SimpleVector.h.

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

8.6.3.9 double HepMC::FourVector::t () const [inline]

return t

Definition at line 78 of file SimpleVector.h.

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

8.6.3.10 double HepMC::FourVector::m2 () const [inline]

Invariant mass squared.

Referenced by main().

8.6.3.11 double HepMC::FourVector::m () const [inline]

Invariant mass. If **m2()** (p. 42) is negative then -sqrt(-m2()) is returned.

Referenced by main().

8.6.3.12 double HepMC::FourVector::perp2 () const [inline]

Transverse component of the spatial vector squared.

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

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

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

Magnitude of the spatial vector.

8.6.3.15 double HepMC::FourVector::theta () const [inline]

The polar angle.

8.6.3.16 double HepMC::FourVector::phi () const [inline]

The azimuth angle.

8.6.3.17 double HepMC::FourVector::rho () const [inline]

spatial vector component magnitude

8.6.3.18 FourVector& HepMC::FourVector::operator= (const FourVector &) [inline]

make a copy

8.6.3.19 bool HepMC::FourVector::operator== (const FourVector &) const [inline]

equality

8.6.3.20 bool HepMC::FourVector::operator!= (const FourVector &) const [inline]

inequality

8.6.3.21 double HepMC::FourVector::pseudoRapidity () const [inline]

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

Referenced by main().

8.6.3.22 double HepMC::FourVector::eta () const [inline]

Pseudorapidity (of the space part).

Referenced by main().

8.6.3.23 void HepMC::FourVector::set (double *x*, double *y*, double *z*, double *t*) [inline]

set x, y, z, and t

Referenced by main().

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

set x

Definition at line 103 of file SimpleVector.h.

Referenced by main().

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

set y

Definition at line 104 of file SimpleVector.h.

Referenced by main().

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

set z

Definition at line 105 of file SimpleVector.h.

Referenced by main().

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

set t

Definition at line 106 of file SimpleVector.h.

Referenced by main().

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

set px

Definition at line 108 of file SimpleVector.h.

Referenced by main().

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

set py

Definition at line 109 of file SimpleVector.h.

Referenced by main().

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

set pz

Definition at line 110 of file SimpleVector.h.

Referenced by main().

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

set E

Definition at line 111 of file SimpleVector.h.

Referenced by main().

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

- **SimpleVector.h**

8.7 HepMC::GenEvent Class Reference

The **GenEvent** (p. 46) 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 >())

  *default constructor creates null pointers to **HeavyIon** (p. 112) and **PdfInfo** (p. 204)*

- **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)

  *explicit constructor that takes **HeavyIon** (p. 112) and **PdfInfo** (p. 204)*

- **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
- **WeightContainer** & **weights** ()
  direct access to ***WeightContainer*** (p. *221*)
- const **WeightContainer** & **weights** () const
  direct access to ***WeightContainer*** (p. *221*)
- **HeavyIon** *const **heavy_ion** () const
  access the ***HeavyIon*** (p. *112*) *container if it exists*
- **HeavyIon** * **heavy_ion** ()
- **PdfInfo** *const **pdf_info** () const
  access the ***PdfInfo*** (p. *204*) *container if it exists*
- **PdfInfo** * **pdf_info** ()
- std::vector< long > **random_states** () const
  vector of integers containing information about the random state
- 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_heavy_ion** (const **HeavyIon** &ion)
  *provide a pointer to the **HeavyIon** (p. 112) container*
- void **set_pdf_info** (const **PdfInfo** &**p**)
  *provide a pointer to the **PdfInfo** (p. 204) container*
- 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
- 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
- **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=0)
  *set the barcode - intended for use by **GenParticle** (p. 77)*
- bool **set_barcode** (**GenVertex** *v, int suggested_barcode=0)
  *set the barcode - intended for use by **GenVertex** (p. 87)*
- void **remove_barcode** (**GenParticle** *p)
  *intended for use by **GenParticle** (p. 77)*
- void **remove_barcode** (**GenVertex** *v)
  *intended for use by **GenVertex** (p. 87)*
- void **delete_all_vertices** ()
  delete all vertices owned by this event

Static Protected Member Functions

- static unsigned int **counter** ()
  *num **GenEvent** (p. 46) objects in memory*

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

8.7.1 Detailed Description

The **GenEvent** (p. 46) class is the core of **HepMC** (p. 19).

HepMC::GenEvent (p. 46) contains information about generated particles. **GenEvent** (p. 46) 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_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, **example_PythiaParticle.cc**, and **example_UsingIterators.cc**.

Definition at line 142 of file GenEvent.h.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 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 >())

default constructor creates null pointers to **HeavyIon** (p. 112) and **PdfInfo** (p. 204)

8.7.2.2 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*)

explicit constructor that takes **HeavyIon** (p. 112) and **PdfInfo** (p. 204)

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

deep copy

deep copy

deep - makes a copy of all vertices!

Definition at line 75 of file GenEvent.cc.

References add_vertex(), alphaQCD(), alphaQED(), beam_particles(), event_number(), event_scale(), GenParticle, GenVertex, mpi(), p, particles_begin(), particles_end(), random_states(), set_alphaQCD(), set_alphaQED(), set_beam_particles(), set_event_number(), set_event_scale(), set_mpi(), set_random_states(), set_signal_process_id(), set_signal_process_vertex(), signal_process_id(), signal_process_vertex(), v, vertices_begin(), vertices_end(), and weights().

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

deletes all vertices/particles in this evt

Deep destructor. deletes all vertices/particles in this evt

Definition at line 170 of file GenEvent.cc.

References delete_all_vertices().

8.7.3 Member Function Documentation

8.7.3.1 GenEvent & HepMC::GenEvent::operator= (const GenEvent & *inevent*)

make a deep copy

best practices implementation

Definition at line 181 of file GenEvent.cc.

References swap().

8.7.3.2 void HepMC::GenEvent::swap (GenEvent & *other*)

swap

Definition at line 150 of file GenEvent.cc.

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

Referenced by operator=().

8.7.3.3 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, and **example_MyHerwig.cc**.

Definition at line 189 of file GenEvent.cc.

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

Referenced by main().

8.7.3.4 void HepMC::GenEvent::print_version (std::ostream & *ostr* = `std::cout`) const

dumps release version to ostr

Definition at line 244 of file GenEvent.cc.

References HepMC::writeVersion().

8.7.3.5 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).

Definition at line 618 of file GenEvent.h.

8.7.3.6 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).

Definition at line 638 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), and HepMC::IO_Ascii::fill_next_event().

8.7.3.7 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 522 of file GenEvent.h.

Referenced by GenEvent(), print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_-AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

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

event number

Examples:

example_EventSelection.cc, and **example_MyPythiaRead.cc**.

Definition at line 525 of file GenEvent.h.

Referenced by GenEvent(), main(), print(), HepMC::IO_HEPEVT::write_event(), HepMC::IO_-ExtendedAscii::write_event(), HepMC::IO_AsciiParticles::write_event(), and HepMC::IO_-Ascii::write_event().

8.7.3.9 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 529 of file GenEvent.h.

Referenced by GenEvent(), and HepMC::IO_ExtendedAscii::write_event().

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

energy scale, see hep-ph/0109068

Definition at line 531 of file GenEvent.h.

Referenced by GenEvent(), print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_-AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

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

QCD coupling, see hep-ph/0109068.

Definition at line 533 of file GenEvent.h.

Referenced by GenEvent(), print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_-AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

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

QED coupling, see hep-ph/0109068

Definition at line 535 of file GenEvent.h.

Referenced by GenEvent(), print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_-AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

8.7.3.13 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 537 of file GenEvent.h.

Referenced by GenEvent(), print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_-AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

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

test to see if we have two valid beam particles

Definition at line 483 of file GenEvent.cc.

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

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

pair of pointers to the two incoming beam particles

Definition at line 659 of file GenEvent.h.

Referenced by GenEvent(), print(), and HepMC::IO_ExtendedAscii::write_event().

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

direct access to **WeightContainer** (p. 221)

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 542 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_-event(), GenEvent(), print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_Ascii-Particles::write_event(), and HepMC::IO_Ascii::write_event().

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

direct access to **WeightContainer** (p. 221)

Definition at line 544 of file GenEvent.h.

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

access the **HeavyIon** (p. 112) container if it exists

Definition at line 547 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::write_event().

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

Definition at line 550 of file GenEvent.h.

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

access the **PdfInfo** (p. 204) container if it exists

Definition at line 553 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::write_event().

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

Definition at line 556 of file GenEvent.h.

8.7.3.22 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 564 of file GenEvent.h.

Referenced by GenEvent(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

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

set unique signal process id

Examples:

> **example_MyHerwig.cc**, **example_MyPythia.cc**, **example_MyPythiaRead.cc**, and **example_PythiaParticle.cc**.

Definition at line 567 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), GenEvent(), and main().

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

set event number

Examples:

example_MyHerwig.cc, **example_MyPythia.cc**, **example_MyPythiaRead.cc**, and **example_PythiaParticle.cc**.

Definition at line 570 of file GenEvent.h.

Referenced by HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), GenEvent(), and main().

8.7.3.25 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**, and **example_MyPythiaWithEventSelection.cc**.

Definition at line 574 of file GenEvent.h.

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

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

set energy scale

Definition at line 578 of file GenEvent.h.

Referenced by GenEvent().

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

set QCD coupling

Definition at line 580 of file GenEvent.h.

Referenced by GenEvent().

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

set QED coupling

Definition at line 582 of file GenEvent.h.

Referenced by GenEvent().

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

set pointer to the vertex containing the signal process

Examples:

example_BuildEventFromScratch.cc.

Definition at line 584 of file GenEvent.h.

References add_vertex().

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

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

set incoming beam particles

construct the beam particle information using pointers to **GenParticle** (p. 77) returns false if either GenParticle* is null

Definition at line 501 of file GenEvent.cc.

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

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

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

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

provide random state information

Definition at line 595 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), and GenEvent().

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

provide a pointer to the **HeavyIon** (p. 112) container

Definition at line 589 of file GenEvent.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event().

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

provide a pointer to the **PdfInfo** (p. 204) container

Definition at line 592 of file GenEvent.h.

References p.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event().

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

how many particle barcodes exist?

Definition at line 645 of file GenEvent.h.

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

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

return true if there are no particle barcodes

Definition at line 648 of file GenEvent.h.

Referenced by delete_all_vertices().

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

how many vertex barcodes exist?

Definition at line 651 of file GenEvent.h.

Referenced by print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

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

return true if there are no vertex barcodes

Definition at line 654 of file GenEvent.h.

Referenced by delete_all_vertices().

8.7.3.39 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.

Definition at line 250 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_HERWIG::build_production_vertex(), HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), GenEvent(), main(), and set_signal_process_vertex().

8.7.3.40 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 273 of file GenEvent.cc.

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

Referenced by add_vertex().

8.7.3.41 void HepMC::GenEvent::clear ()

empties the entire event

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

Definition at line 281 of file GenEvent.cc.

References HepMC::GenParticle::counter(), HepMC::GenVertex::counter(), and delete_all_vertices().

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

begin vertex iteration

Examples:

example_UsingIterators.cc.

Definition at line 294 of file GenEvent.h.

Referenced by GenEvent(), main(), HepMC::IO_HEPEVT::write_event(), HepMC::IO_ExtendedAscii::write_event(), and HepMC::IO_Ascii::write_event().

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

end vertex iteration

Examples:

example_UsingIterators.cc.

Definition at line 298 of file GenEvent.h.

Referenced by GenEvent(), main(), print(), HepMC::IO_HEPEVT::write_event(), HepMC::IO_ExtendedAscii::write_event(), and HepMC::IO_Ascii::write_event().

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

begin vertex iteration

Definition at line 351 of file GenEvent.h.

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

end vertex iteration

Definition at line 355 of file GenEvent.h.

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

begin particle iteration

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_UsingIterators.cc**.

Definition at line 413 of file GenEvent.h.

Referenced by GenEvent(), main(), IsGoodEventMyPythia::operator()(), IsGoodEvent::operator()(), valid_beam_particles(), and HepMC::IO_AsciiParticles::write_event().

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

end particle iteration

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_UsingIterators.cc**.

Definition at line 417 of file GenEvent.h.

Referenced by GenEvent(), main(), IsGoodEventMyPythia::operator()(), IsGoodEvent::operator()(), valid_beam_particles(), and HepMC::IO_AsciiParticles::write_event().

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

begin particle iteration

Definition at line 466 of file GenEvent.h.

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

end particle iteration

Definition at line 470 of file GenEvent.h.

8.7.3.50 bool HepMC::GenEvent::set_barcode (GenParticle * *p*, int *suggested_barcode* = 0) [protected]

set the barcode - intended for use by **GenParticle** (p. 77)

Definition at line 345 of file GenEvent.cc.

References p.

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

8.7.3.51 bool HepMC::GenEvent::set_barcode (GenVertex * *v*, int *suggested_barcode* = 0) [protected]

set the barcode - intended for use by **GenVertex** (p. 87)

Definition at line 416 of file GenEvent.cc.

References v.

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

intended for use by **GenParticle** (p. 77)

Definition at line 599 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().

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

intended for use by **GenVertex** (p. 87)

Definition at line 602 of file GenEvent.h.

References v.

8.7.3.54 unsigned int HepMC::GenEvent::counter () [static, protected]

num **GenEvent** (p. 46) objects in memory

Definition at line 517 of file GenEvent.cc.

Referenced by print().

8.7.3.55 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 desctructors are automatically deleting their particles.

Definition at line 314 of file GenEvent.cc.

References HepMC::GenParticle::counter(), HepMC::GenVertex::counter(), particles_empty(), and vertices_empty().

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

8.7.4 Friends And Related Function Documentation

8.7.4.1 friend class GenParticle [friend]

Definition at line 143 of file GenEvent.h.

Referenced by GenEvent().

8.7.4.2 friend class GenVertex [friend]

Definition at line 144 of file GenEvent.h.

Referenced by GenEvent().

8.7.4.3 friend class vertex_const_iterator [friend]

Definition at line 292 of file GenEvent.h.

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

8.7.4.4 friend class vertex_iterator [friend]

Definition at line 349 of file GenEvent.h.

8.7.4.5 friend class particle_const_iterator [friend]

Definition at line 411 of file GenEvent.h.

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

8.7.4.6 friend class particle_iterator [friend]

Definition at line 464 of file GenEvent.h.

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

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

8.8 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. 77)*
- **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. 77) map*

8.8.1 Detailed Description

const particle iterator

HepMC::GenEvent::particle_const_iterator (p. 63) is used to iterate over all particles in the event.

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, and **example_MyPythiaWithEventSelection.cc**.

Definition at line 375 of file GenEvent.h.

8.8.2 Constructor & Destructor Documentation

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

iterate over particles

Definition at line 380 of file GenEvent.h.

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

Definition at line 383 of file GenEvent.h.

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

copy constructor

Definition at line 385 of file GenEvent.h.

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

Definition at line 387 of file GenEvent.h.

8.8.3 Member Function Documentation

8.8.3.1 particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator= (const particle_const_iterator & *i*) [inline]

make a copy

Definition at line 389 of file GenEvent.h.

References m_map_iterator.

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

return a pointer to **GenParticle** (p. 77)

Definition at line 393 of file GenEvent.h.

References m_map_iterator.

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

Pre-fix increment.

Definition at line 396 of file GenEvent.h.

References m_map_iterator.

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

Post-fix increment.

Definition at line 399 of file GenEvent.h.

8.8.3.5 bool HepMC::GenEvent::particle_const_iterator::operator== (const particle_const_iterator & *a*) const [inline]

equality

Definition at line 402 of file GenEvent.h.

References m_map_iterator.

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

inequality

Definition at line 405 of file GenEvent.h.

References m_map_iterator.

8.8.4 Member Data Documentation

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

const iterator to the **GenParticle** (p. 77) map

Definition at line 409 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**

8.9 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. 77)*
- **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. 77) map*

8.9.1 Detailed Description

non-const particle iterator

HepMC::GenEvent::particle_iterator (p. 66) is used to iterate over all particles in the event.

Examples:

example_UsingIterators.cc.

Definition at line 426 of file GenEvent.h.

8.9.2 Constructor & Destructor Documentation

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

iterate over particles

Definition at line 431 of file GenEvent.h.

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

Definition at line 433 of file GenEvent.h.

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

copy constructor

Definition at line 435 of file GenEvent.h.

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

Definition at line 436 of file GenEvent.h.

8.9.3 Member Function Documentation

8.9.3.1 particle_iterator& HepMC::GenEvent::particle_iterator::operator= (const particle_iterator & *i*) [inline]

make a copy

Definition at line 438 of file GenEvent.h.

References m_map_iterator.

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

const particle iterator

Definition at line 443 of file GenEvent.h.

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

8.9.3.3 GenParticle* HepMC::GenEvent::particle_iterator::operator * (void) const [inline]

return pointer to **GenParticle** (p. 77)

Definition at line 446 of file GenEvent.h.

References m_map_iterator.

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

Pre-fix increment.

Definition at line 449 of file GenEvent.h.

References m_map_iterator.

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

Post-fix increment.

Definition at line 452 of file GenEvent.h.

8.9.3.6 bool HepMC::GenEvent::particle_iterator::operator== (const particle_iterator & *a*) const [inline]

equality

Definition at line 455 of file GenEvent.h.

References m_map_iterator.

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

inequality

Definition at line 458 of file GenEvent.h.

References m_map_iterator.

8.9.4 Member Data Documentation

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

iterator for **GenParticle** (p. 77) map

Definition at line 462 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

8.10 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. 87)
- **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

8.10.1 Detailed Description

const vertex iterator

HepMC::GenEvent::vertex_const_iterator (p. 70) is used to iterate over all vertices in the event.

Definition at line 256 of file GenEvent.h.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 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 261 of file GenEvent.h.

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

Definition at line 265 of file GenEvent.h.

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

copy constructor

Definition at line 267 of file GenEvent.h.

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

Definition at line 269 of file GenEvent.h.

8.10.3 Member Function Documentation

8.10.3.1 vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator= (const vertex_const_iterator & *i*) [inline]

make a copy

Definition at line 271 of file GenEvent.h.

References m_map_iterator.

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

return a pointer to a **GenVertex** (p. 87)

Definition at line 274 of file GenEvent.h.

References m_map_iterator.

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

Pre-fix increment.

Definition at line 276 of file GenEvent.h.

References m_map_iterator.

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

Post-fix increment.

Definition at line 279 of file GenEvent.h.

8.10.3.5 bool HepMC::GenEvent::vertex_const_iterator::operator== (const vertex_const_iterator & *a*) const [inline]

equality

Definition at line 282 of file GenEvent.h.

References m_map_iterator.

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

inequality

Definition at line 285 of file GenEvent.h.

References m_map_iterator.

8.10.4 Member Data Documentation

8.10.4.1 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 290 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**

8.11 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. 87)*

- **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

8.11.1 Detailed Description

non-const vertex iterator

HepMC::GenEvent::vertex_iterator (p. 73) is used to iterate over all vertices in the event.

Examples:

example_UsingIterators.cc.

Definition at line 308 of file GenEvent.h.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 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 313 of file GenEvent.h.

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

Definition at line 317 of file GenEvent.h.

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

copy constructor

Definition at line 319 of file GenEvent.h.

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

Definition at line 320 of file GenEvent.h.

8.11.3 Member Function Documentation

8.11.3.1 vertex_iterator& HepMC::GenEvent::vertex_iterator::operator= (const vertex_iterator & *i*) [inline]

make a copy

Definition at line 322 of file GenEvent.h.

References m_map_iterator.

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

const vertex iterator

Definition at line 327 of file GenEvent.h.

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

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

return a pointer to a **GenVertex** (p. 87)

Definition at line 330 of file GenEvent.h.

References m_map_iterator.

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

Pre-fix increment.

Definition at line 333 of file GenEvent.h.

References m_map_iterator.

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

Post-fix increment.

Definition at line 336 of file GenEvent.h.

8.11.3.6 bool HepMC::GenEvent::vertex_iterator::operator== (const vertex_iterator & *a*) const [inline]

equality

Definition at line 339 of file GenEvent.h.

References m_map_iterator.

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

inequality

Definition at line 342 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> >::iterator HepMC::GenEvent::vertex_iterator::m_map_iterator [protected]

iterator to the vertex map

Definition at line 347 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**

8.12 HepMC::GenParticle Class Reference

The **GenParticle** (p. 77) 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** &itsflow=**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
- **FourVector momentum** () const
  standard 4 momentum
- int **pdg_id** () const
  particle ID
- int **status** () const
  HEPEVT decay status.
- **Flow flow** () const
  particle flow
- int **flow** (int code_index) const
  particle flow index

- **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. 83) is included for backwards compatibility with* ***CLHEP*** *(p. 17)* ***HepMC*** *(p. 19)*
- int **barcode** () const
  particle barcode
- hepmc_uint64_t **serialnumber** () const
  used by GenParticleComparison
- 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. 84) 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

- void **set_end_vertex_** (**GenVertex** *decayvertex=0)

  set decay vertex

- void **set_barcode_** (int the_bar_code)

  *for use by **GenEvent** (p. 46) only*

Static Protected Member Functions

- static unsigned int **counter** ()

  temporary for debugging

Friends

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

  print particle

8.12.1 Detailed Description

The **GenParticle** (p. 77) class contains information about generated particles.

HepMC::GenParticle (p. 77) contains momentum, generated mass, particle ID, decay status, flow, polarization, pointers to production and decay vertices and a unique barcode identfier.

Examples:

example_BuildEventFromScratch.cc, and **example_UsingIterators.cc**.

Definition at line 55 of file GenParticle.h.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 HepMC::GenParticle::GenParticle (void)

default constructor

Definition at line 14 of file GenParticle.cc.

8.12.2.2 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().

8.12.2.3 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 38 of file GenParticle.cc.

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

8.12.2.4 HepMC::GenParticle::~GenParticle () [virtual]

Definition at line 60 of file GenParticle.cc.

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

8.12.3 Member Function Documentation

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

swap

Definition at line 65 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_serialnumber, m_status, HepMC::Polarization::swap(), HepMC::Flow::swap(), and HepMC::FourVector::swap().

Referenced by operator=().

8.12.3.2 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 80 of file GenParticle.cc.

References swap().

8.12.3.3 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 92 of file GenParticle.cc.

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

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

check for inequality

Definition at line 105 of file GenParticle.cc.

8.12.3.5 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 109 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().

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

conversion operator

Definition at line 175 of file GenParticle.h.

8.12.3.7 FourVector HepMC::GenParticle::momentum () const [inline]

standard 4 momentum

Definition at line 178 of file GenParticle.h.

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

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

particle ID

Definition at line 181 of file GenParticle.h.

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

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

HEPEVT decay status.

Definition at line 183 of file GenParticle.h.

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

8.12.3.10 Flow HepMC::GenParticle::flow () const [inline]

particle flow

Definition at line 190 of file GenParticle.h.

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

particle flow index

Definition at line 192 of file GenParticle.h.

References HepMC::Flow::icode().

8.12.3.12 Polarization HepMC::GenParticle::polarization () const [inline]

polarization information

Definition at line 195 of file GenParticle.h.

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

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

pointer to the production vertex

Definition at line 185 of file GenParticle.h.

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

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

pointer to the decay vertex

Definition at line 188 of file GenParticle.h.

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

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

pointer to the event that owns this particle

Definition at line 126 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().

8.12.3.16 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. 82) returns that value. If the generated mass has not been set, then **generated_mass()** (p. 82) returns the mass calculated from the momentum 4 vector.

Definition at line 239 of file GenParticle.cc.

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

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

generatedMass() (p. 83) is included for backwards compatibility with **CLHEP** (p. 17) **HepMC** (p. 19)

Definition at line 116 of file GenParticle.h.

References generated_mass().

8.12.3.18 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.

Definition at line 219 of file GenParticle.h.

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

8.12.3.19 hepmc_uint64_t HepMC::GenParticle::serialnumber () const

used by GenParticleComparison

Definition at line 187 of file GenParticle.cc.

8.12.3.20 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 156 of file GenParticle.cc.

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

Referenced by GenParticle().

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

set standard 4 momentum

Definition at line 198 of file GenParticle.h.

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

set particle ID

Definition at line 201 of file GenParticle.h.

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

set decay status

Definition at line 203 of file GenParticle.h.

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

set particle flow

Definition at line 205 of file GenParticle.h.

Referenced by GenParticle().

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

set particle flow index

Definition at line 207 of file GenParticle.h.

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

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

set polarization

Definition at line 216 of file GenParticle.h.

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

define the actual generated mass

If you do not call **set_generated_mass()** (p. 84), then **generated_mass()** (p. 82) will simply return the mass calculated from **momentum()** (p. 81)

Definition at line 243 of file GenParticle.cc.

Referenced by setGeneratedMass().

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

setGeneratedMass() (p. 84) is included for backwards compatibility with **CLHEP** (p. 17) **HepMC** (p. 19)

Definition at line 142 of file GenParticle.h.

References set_generated_mass().

8.12.3.29 unsigned int HepMC::GenParticle::counter () [static, protected]

temporary for debugging

Definition at line 184 of file GenParticle.cc.

Referenced by HepMC::GenEvent::clear(), HepMC::GenEvent::delete_all_vertices(), and HepMC::GenEvent::print().

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

set production vertex

Definition at line 132 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().

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

set decay vertex

Definition at line 145 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().

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

for use by **GenEvent** (p. 46) only

Definition at line 221 of file GenParticle.h.

Referenced by suggest_barcode().

8.12.4 Friends And Related Function Documentation

8.12.4.1 friend class GenVertex [friend]

Definition at line 57 of file GenParticle.h.

8.12.4.2 friend class GenEvent [friend]

Definition at line 58 of file GenParticle.h.

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

print particle

Definition at line 195 of file GenParticle.cc.

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

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

8.13 HepMC::GenVertex Class Reference

GenVertex (p. 87) 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 **WeightContainer** &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
- **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
- 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

Static Protected Member Functions

- static unsigned int **counter** ()

  temporary for debugging

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

8.13.1 Detailed Description

GenVertex (p. 87) contains information about decay vertices.

HepMC::GenVertex (p. 87) contains the position in space and time of a decay. It also contains lists of incoming and outgoing particles.

Examples:

example_BuildEventFromScratch.cc.

Definition at line 47 of file GenVertex.h.

8.13.2 Member Typedef Documentation

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

const iterator for incoming particles

Definition at line 130 of file GenVertex.h.

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

const iterator for outgoing particles

Definition at line 133 of file GenVertex.h.

8.13.3 Constructor & Destructor Documentation

8.13.3.1 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.

8.13.3.2 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 22 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().

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

Definition at line 62 of file GenVertex.cc.

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

8.13.4 Member Function Documentation

8.13.4.1 void HepMC::GenVertex::swap (GenVertex & *other*)

swap

Definition at line 70 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=().

8.13.4.2 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 81 of file GenVertex.cc.

References swap().

8.13.4.3 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 102 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().

8.13.4.4 bool HepMC::GenVertex::operator!= (const GenVertex & *a*) const

inequality

Definition at line 139 of file GenVertex.cc.

8.13.4.5 void HepMC::GenVertex::print (std::ostream & *ostr* = `std::cout`) const

print vertex information

Definition at line 144 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::WeightContainer::size(), HepMC::FourVector::t(), weights(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

Referenced by HepMC::IO_HERWIG::build_production_vertex().

8.13.4.6 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 | {p_in} - {p_out} |

Definition at line 252 of file GenVertex.cc.

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

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

add incoming particle

Examples:

example_BuildEventFromScratch.cc.

Definition at line 272 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_HERWIG::build_production_vertex(), HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), GenVertex(), and main().

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

add outgoing particle

Examples:

example_BuildEventFromScratch.cc.

Definition at line 283 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_HERWIG::build_production_vertex(), HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), GenVertex(), and main().

8.13.4.9 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 294 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_().

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

conversion operator

Definition at line 365 of file GenVertex.h.

References position().

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

conversion operator

Definition at line 367 of file GenVertex.h.

References point3d().

8.13.4.12 GenEvent * HepMC::GenVertex::parent_event () const [inline]

pointer to the event that owns this vertex

Definition at line 371 of file GenVertex.h.

Referenced by HepMC::GenEvent::add_vertex(), HepMC::GenParticle::parent_event(), HepMC::GenEvent::remove_vertex(), suggest_barcode(), and ~GenVertex().

8.13.4.13 ThreeVector HepMC::GenVertex::point3d () const [inline]

vertex position

Definition at line 373 of file GenVertex.h.

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

Referenced by operator HepMC::ThreeVector().

8.13.4.14 FourVector HepMC::GenVertex::position () const [inline]

vertex position and time

Definition at line 369 of file GenVertex.h.

Referenced by HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HERWIG::build_production_vertex(), operator HepMC::FourVector(), HepMC::operator<<(), operator==(), print(), and set_position().

8.13.4.15 void HepMC::GenVertex::set_position (const FourVector & *position* = FourVector(0, 0, 0, 0)) [inline]

set vertex position and time

Definition at line 387 of file GenVertex.h.

References position().

Referenced by HepMC::IO_HERWIG::build_end_vertex(), and HepMC::IO_HERWIG::build_production_vertex().

8.13.4.16 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. 95)

Definition at line 377 of file GenVertex.h.

Referenced by print().

8.13.4.17 void HepMC::GenVertex::set_id (int *id*) [inline]

set vertex ID

Definition at line 391 of file GenVertex.h.

8.13.4.18 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.

Definition at line 379 of file GenVertex.h.

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

8.13.4.19 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 362 of file GenVertex.cc.

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

Referenced by GenVertex().

8.13.4.20 WeightContainer & HepMC::GenVertex::weights () [inline]

direct access to the weights container is allowed.

Definition at line 382 of file GenVertex.h.

Referenced by print().

8.13.4.21 const WeightContainer & HepMC::GenVertex::weights () const [inline]

const direct access to the weights container

Definition at line 384 of file GenVertex.h.

8.13.4.22 GenVertex::particles_in_const_iterator HepMC::GenVertex::particles_in_const_begin () const [inline]

begin iteration of incoming particles

Definition at line 398 of file GenVertex.h.

Referenced by check_momentum_conservation(), GenVertex(), operator==(), print(), and set_parent_event_().

8.13.4.23 GenVertex::particles_in_const_iterator HepMC::GenVertex::particles_in_const_end () const [inline]

end iteration of incoming particles

Definition at line 403 of file GenVertex.h.

Referenced by check_momentum_conservation(), GenVertex(), operator==(), print(), and set_parent_event_().

8.13.4.24 GenVertex::particles_out_const_iterator HepMC::GenVertex::particles_out_const_begin () const [inline]

begin iteration of outgoing particles

Definition at line 408 of file GenVertex.h.

Referenced by check_momentum_conservation(), GenVertex(), operator==(), print(), and set_parent_event_().

8.13.4.25 GenVertex::particles_out_const_iterator HepMC::GenVertex::particles_out_const_end () const [inline]

end iteration of outgoing particles

Definition at line 413 of file GenVertex.h.

Referenced by check_momentum_conservation(), GenVertex(), operator==(), print(), and set_parent_event_().

8.13.4.26 int HepMC::GenVertex::particles_in_size () const [inline]

number of incoming particles

Definition at line 417 of file GenVertex.h.

Referenced by operator==().

8.13.4.27 int HepMC::GenVertex::particles_out_size () const [inline]

number of outgoing particles

Definition at line 421 of file GenVertex.h.

Referenced by operator==().

8.13.4.28 unsigned int HepMC::GenVertex::counter () [static, protected]

temporary for debugging

Definition at line 421 of file GenVertex.cc.

Referenced by HepMC::GenEvent::clear(), HepMC::GenEvent::delete_all_vertices(), and HepMC::GenEvent::print().

8.13.4.29 void HepMC::GenVertex::set_parent_event_ (GenEvent * *evt*) [protected]

set parent event

only the **GenEvent** (p. 46) (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 387 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().

8.13.4.30 void HepMC::GenVertex::set_barcode_ (int *the_bar_code*) [inline, protected]

set identifier

Definition at line 380 of file GenVertex.h.

Referenced by suggest_barcode().

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

size

Definition at line 572 of file GenVertex.cc.

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

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

begin range

Definition at line 439 of file GenVertex.h.

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

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

end range

Definition at line 444 of file GenVertex.h.

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

8.13.4.34 GenVertex::vertex_iterator HepMC::GenVertex::vertices_begin (IteratorRange *range* = relatives) [inline]

begin vertex range

Definition at line 467 of file GenVertex.h.

References vertex_iterator.

8.13.4.35 GenVertex::vertex_iterator HepMC::GenVertex::vertices_end (IteratorRange) [inline]

end vertex range

Definition at line 473 of file GenVertex.h.

References vertex_iterator.

8.13.4.36 GenVertex::particle_iterator HepMC::GenVertex::particles_begin (IteratorRange *range* = relatives) [inline]

begin particle range

Definition at line 488 of file GenVertex.h.

References particle_iterator.

8.13.4.37 GenVertex::particle_iterator HepMC::GenVertex::particles_end (IteratorRange) [inline]

end particle range

Definition at line 493 of file GenVertex.h.

References particle_iterator.

8.13.4.38 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 328 of file GenVertex.cc.

Referenced by ~GenVertex().

8.13.4.39 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 316 of file GenVertex.cc.

References HepMC::already_in_vector().

Referenced by add_particle_in(), and remove_particle().

8.13.4.40 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 322 of file GenVertex.cc.

References HepMC::already_in_vector().

Referenced by add_particle_out(), and remove_particle().

8.13.5 Friends And Related Function Documentation

8.13.5.1 friend class GenEvent [friend]

Definition at line 51 of file GenVertex.h.

8.13.5.2 friend class edge_iterator [friend]

Definition at line 205 of file GenVertex.h.

8.13.5.3 friend class vertex_iterator [friend]

Definition at line 284 of file GenVertex.h.

Referenced by vertices_begin(), and vertices_end().

8.13.5.4 friend class particle_iterator [friend]

Definition at line 332 of file GenVertex.h.

Referenced by particles_begin(), and particles_end().

8.13.5.5 std::ostream& operator<< (std::ostream & *ostr*, const GenVertex & *vtx*) [friend]

print vertex information

Definition at line 429 of file GenVertex.cc.

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

- GenVertex.h
- GenVertex.cc

8.14 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

8.14.1 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 171 of file GenVertex.h.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 HepMC::GenVertex::edge_iterator::edge_iterator ()

Definition at line 451 of file GenVertex.cc.

8.14.2.2 HepMC::GenVertex::edge_iterator::edge_iterator (const GenVertex & *vtx*, IteratorRange *range* = `family`)

used to set limits on the iteration

Definition at line 455 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.

8.14.2.3 HepMC::GenVertex::edge_iterator::edge_iterator (const edge_iterator & *p*)

copy

Definition at line 506 of file GenVertex.cc.

References p.

8.14.2.4 HepMC::GenVertex::edge_iterator::~edge_iterator () `[virtual]`

Definition at line 510 of file GenVertex.cc.

8.14.3 Member Function Documentation

8.14.3.1 GenVertex::edge_iterator & HepMC::GenVertex::edge_iterator::operator= (const edge_iterator & *p*)

make a copy

Definition at line 512 of file GenVertex.cc.

References p.

8.14.3.2 GenParticle * HepMC::GenVertex::edge_iterator::operator * (void) const

return a pointer to a particle

Definition at line 522 of file GenVertex.cc.

8.14.3.3 GenVertex::edge_iterator & HepMC::GenVertex::edge_iterator::operator++ (void)

Pre-fix increment.

Definition at line 527 of file GenVertex.cc.

References HepMC::family, HepMC::GenVertex::m_particles_in, HepMC::GenVertex::m_particles_out, and HepMC::parents.

8.14.3.4 GenVertex::edge_iterator HepMC::GenVertex::edge_iterator::operator++ (int)

Post-fix increment.

Definition at line 555 of file GenVertex.cc.

8.14.3.5 bool HepMC::GenVertex::edge_iterator::operator== (const edge_iterator & *a*) const [inline]

equality

Definition at line 425 of file GenVertex.h.

8.14.3.6 bool HepMC::GenVertex::edge_iterator::operator!= (const edge_iterator & *a*) const [inline]

inequality

Definition at line 430 of file GenVertex.h.

8.14.3.7 bool HepMC::GenVertex::edge_iterator::is_parent () const

true if parent of root vtx

Definition at line 562 of file GenVertex.cc.

Referenced by HepMC::GenVertex::particle_iterator::advance_to_first_(), and HepMC::GenVertex::vertex_iterator::follow_edge_().

8.14.3.8 bool HepMC::GenVertex::edge_iterator::is_child () const

true if child of root vtx

Definition at line 567 of file GenVertex.cc.

8.14.3.9 const GenVertex * HepMC::GenVertex::edge_iterator::vertex_root () const [inline]

root vertex of this iteration

Definition at line 435 of file GenVertex.h.

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

- GenVertex.h
- GenVertex.cc

8.15 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

8.15.1 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.

Definition at line 305 of file GenVertex.h.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 HepMC::GenVertex::particle_iterator::particle_iterator ()

Definition at line 815 of file GenVertex.cc.

8.15.2.2 HepMC::GenVertex::particle_iterator::particle_iterator (GenVertex & *vertex_root*, IteratorRange *range*)

used to set limits on the iteration

Definition at line 817 of file GenVertex.cc.

References advance_to_first_(), HepMC::family, and HepMC::GenVertex::vertex_iterator::range().

8.15.2.3 HepMC::GenVertex::particle_iterator::particle_iterator (const particle_iterator &)

copy

Definition at line 831 of file GenVertex.cc.

8.15.2.4 HepMC::GenVertex::particle_iterator::~particle_iterator () [virtual]

Definition at line 836 of file GenVertex.cc.

8.15.3 Member Function Documentation

8.15.3.1 GenVertex::particle_iterator & HepMC::GenVertex::particle_iterator::operator= (const particle_iterator &)

make a copy

Definition at line 839 of file GenVertex.cc.

References m_edge, and m_vertex_iterator.

8.15.3.2 GenParticle * HepMC::GenVertex::particle_iterator::operator * (void) const

return a pointer to a particle

Definition at line 846 of file GenVertex.cc.

8.15.3.3 GenVertex::particle_iterator & HepMC::GenVertex::particle_iterator::operator++ (void)

Pre-fix increment.

Definition at line 851 of file GenVertex.cc.

References advance_to_first_(), and HepMC::GenVertex::vertex_iterator::range().

8.15.3.4 GenVertex::particle_iterator HepMC::GenVertex::particle_iterator::operator++ (int)

Post-fix increment.

Definition at line 870 of file GenVertex.cc.

8.15.3.5 bool HepMC::GenVertex::particle_iterator::operator== (const particle_iterator &) const [inline]

equality

Definition at line 478 of file GenVertex.h.

8.15.3.6 bool HepMC::GenVertex::particle_iterator::operator!= (const particle_iterator &) const [inline]

inequality

Definition at line 483 of file GenVertex.h.

8.15.3.7 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 877 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().

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

- **GenVertex.h**
- **GenVertex.cc**

8.16 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

8.16.1 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 235 of file GenVertex.h.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 HepMC::GenVertex::vertex_iterator::vertex_iterator ()

Definition at line 584 of file GenVertex.cc.

Referenced by copy_recursive_iterator_(), and follow_edge_().

8.16.2.2 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & *vtx_root*, IteratorRange *range*)

used to set limits on the iteration

Definition at line 589 of file GenVertex.cc.

References HepMC::GenVertex::edges_begin(), HepMC::GenVertex::edges_end(), and follow_edge_().

8.16.2.3 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

8.16.2.4 HepMC::GenVertex::vertex_iterator::vertex_iterator (const vertex_iterator & *v_iter*)

copy

Definition at line 622 of file GenVertex.cc.

8.16.2.5 HepMC::GenVertex::vertex_iterator::~vertex_iterator () [virtual]

Definition at line 629 of file GenVertex.cc.

8.16.3 Member Function Documentation

8.16.3.1 GenVertex::vertex_iterator & HepMC::GenVertex::vertex_iterator::operator= (const vertex_iterator &)

make a copy

Definition at line 634 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.

8.16.3.2 GenVertex * HepMC::GenVertex::vertex_iterator::operator * (void) const

return a pointer to a vertex

Definition at line 671 of file GenVertex.cc.

8.16.3.3 GenVertex::vertex_iterator & HepMC::GenVertex::vertex_iterator::operator++ (void)

Pre-fix increment.

Definition at line 686 of file GenVertex.cc.

References HepMC::GenVertex::edges_end(), and follow_edge_().

8.16.3.4 GenVertex::vertex_iterator HepMC::GenVertex::vertex_iterator::operator++ (int)

Post-fix increment.

Definition at line 728 of file GenVertex.cc.

8.16.3.5 bool HepMC::GenVertex::vertex_iterator::operator== (const vertex_iterator &) const [inline]

equality

Definition at line 449 of file GenVertex.h.

8.16.3.6 bool HepMC::GenVertex::vertex_iterator::operator!= (const vertex_iterator &) const [inline]

inequality

Definition at line 454 of file GenVertex.h.

8.16.3.7 GenVertex * HepMC::GenVertex::vertex_iterator::vertex_root () const [inline]

vertex that this iterator begins from

Definition at line 459 of file GenVertex.h.

8.16.3.8 IteratorRange HepMC::GenVertex::vertex_iterator::range () const [inline]

iterator range

Definition at line 463 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().

8.16.3.9 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.

8.16.3.10 GenVertex * HepMC::GenVertex::vertex_iterator::follow_edge_ () [protected]

non-null if recursive iter. created

Definition at line 758 of file GenVertex.cc.

References HepMC::family, HepMC::GenVertex::edge_iterator::is_parent(), and vertex_iterator().

Referenced by operator++(), and vertex_iterator().

8.16.3.11 void HepMC::GenVertex::vertex_iterator::copy_recursive_iterator_ (const vertex_iterator * *recursive_v_iter*) [protected]

copy recursive iterator

Definition at line 794 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_(), and operator=().

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

- **GenVertex.h**
- **GenVertex.cc**

8.17 HepMC::HeavyIon Class Reference

The **HeavyIon** (p. 112) 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. 112) 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 collisons.
- 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
- 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 collisons
- 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

8.17.1 Detailed Description

The **HeavyIon** (p. 112) class stores information about heavy ions.

HepMC::HeavyIon (p. 112) provides additional information storage for Heavy Ion generators in **GenEvent** (p. 46). Creation and use of this information is optional.

Definition at line 45 of file HeavyIon.h.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 HepMC::HeavyIon::HeavyIon () [inline]

default constructor

Definition at line 51 of file HeavyIon.h.

8.17.2.2 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 168 of file HeavyIon.h.

8.17.2.3 HepMC::HeavyIon::~HeavyIon () [inline]

Definition at line 72 of file HeavyIon.h.

8.17.2.4 HepMC::HeavyIon::HeavyIon (HeavyIon const & *orig*) [inline]

copy constructor

Definition at line 186 of file HeavyIon.h.

8.17.3 Member Function Documentation

8.17.3.1 HeavyIon & HepMC::HeavyIon::operator= (HeavyIon const & *rhs*) [inline]

make a copy

Definition at line 202 of file HeavyIon.h.

References swap().

8.17.3.2 void HepMC::HeavyIon::swap (HeavyIon & *other*) [inline]

swap two **HeavyIon** (p. 112) objects

Definition at line 209 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=().

8.17.3.3 bool HepMC::HeavyIon::operator== (const HeavyIon &) const [inline]

check for equality

equality requires that each member match

Definition at line 226 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().

8.17.3.4 bool HepMC::HeavyIon::operator!= (const HeavyIon &) const [inline]

check for inequality

any nonmatching member generates inequality

Definition at line 244 of file HeavyIon.h.

8.17.3.5 int HepMC::HeavyIon::Ncoll_hard () const [inline]

Number of hard scatterings.

Definition at line 87 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.6 int HepMC::HeavyIon::Npart_proj () const [inline]

Number of projectile participants.

Definition at line 89 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.7 int HepMC::HeavyIon::Npart_targ () const [inline]

Number of target participants.

Definition at line 91 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.8 int HepMC::HeavyIon::Ncoll () const [inline]

Number of NN (nucleon-nucleon) collisions.

Definition at line 93 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.9 int HepMC::HeavyIon::spectator_neutrons () const [inline]

Number of spectator neutrons.

Definition at line 95 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.10 int HepMC::HeavyIon::spectator_protons () const [inline]

Number of spectator protons.

Definition at line 97 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.11 int HepMC::HeavyIon::N_Nwounded_collisions () const [inline]

Number of N-Nwounded collisions.

Definition at line 99 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.12 int HepMC::HeavyIon::Nwounded_N_collisions () const [inline]

Number of Nwounded-N collisons.

Definition at line 101 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.13 int HepMC::HeavyIon::Nwounded_Nwounded_collisions () const [inline]

Number of Nwounded-Nwounded collisions.

Definition at line 103 of file HeavyIon.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.14 float HepMC::HeavyIon::impact_parameter () const [inline]

Impact Parameter(in fm) of collision.
Definition at line 105 of file HeavyIon.h.
Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.15 float HepMC::HeavyIon::event_plane_angle () const [inline]

Azimuthal angle of event plane.
Definition at line 107 of file HeavyIon.h.
Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.16 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 operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.17 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 operator==(), and HepMC::IO_ExtendedAscii::write_heavy_ion().

8.17.3.18 void HepMC::HeavyIon::set_Ncoll_hard (const int & *i*) [inline]

set number of hard scatterings
Definition at line 116 of file HeavyIon.h.

8.17.3.19 void HepMC::HeavyIon::set_Npart_proj (const int & *i*) [inline]

set number of projectile participants
Definition at line 118 of file HeavyIon.h.

8.17.3.20 void HepMC::HeavyIon::set_Npart_targ (const int & *i*) [inline]

set number of target participants
Definition at line 120 of file HeavyIon.h.

8.17.3.21 void HepMC::HeavyIon::set_Ncoll (const int & *i*) [inline]

set number of NN (nucleon-nucleon) collisions

Definition at line 122 of file HeavyIon.h.

8.17.3.22 void HepMC::HeavyIon::set_spectator_neutrons (const int & *i*) [inline]

set number of spectator neutrons

Definition at line 124 of file HeavyIon.h.

8.17.3.23 void HepMC::HeavyIon::set_spectator_protons (const int & *i*) [inline]

set number of spectator protons

Definition at line 126 of file HeavyIon.h.

8.17.3.24 void HepMC::HeavyIon::set_N_Nwounded_collisions (const int & *i*) [inline]

set number of N-Nwounded collisions

Definition at line 128 of file HeavyIon.h.

8.17.3.25 void HepMC::HeavyIon::set_Nwounded_N_collisions (const int & *i*) [inline]

set number of Nwounded-N collisons

Definition at line 130 of file HeavyIon.h.

8.17.3.26 void HepMC::HeavyIon::set_Nwounded_Nwounded_collisions (const int & *i*) [inline]

set number of Nwounded-Nwounded collisions

Definition at line 132 of file HeavyIon.h.

8.17.3.27 void HepMC::HeavyIon::set_impact_parameter (const float & *f*) [inline]

set Impact Parameter in fm

Definition at line 135 of file HeavyIon.h.

8.17.3.28 void HepMC::HeavyIon::set_event_plane_angle (const float & *f*) [inline]

set azimuthal angle of event plane

Definition at line 137 of file HeavyIon.h.

8.17.3.29 void HepMC::HeavyIon::set_eccentricity (const float & *f*) [inline]

set eccentricity of participating nucleons in the transverse plane

Definition at line 139 of file HeavyIon.h.

8.17.3.30 void HepMC::HeavyIon::set_sigma_inel_NN (const float & *f*) [inline]

set nucleon-nucleon inelastic cross-section

Definition at line 141 of file HeavyIon.h.

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

- **HeavyIon.h**

8.18 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

8.18.1 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.

8.18.2 Member Function Documentation

8.18.2.1 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().

8.18.2.2 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().

8.18.2.3 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().

8.18.2.4 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().

8.18.2.5 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().

8.18.2.6 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_HERWIG::build_production_vertex(), check_hepevt_consistency(), HepMC::IO_HERWIG::fill_next_event(), HepMC::IO_HEPEVT::fill_next_event(), and print_hepevt().

8.18.2.7 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().

8.18.2.8 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().

8.18.2.9 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().

8.18.2.10 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(), check_hepevt_consistency(), last_parent(), number_parents(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.18.2.11 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(), check_hepevt_consistency(), number_parents(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.18.2.12 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().

8.18.2.13 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(), check_hepevt_consistency(), last_child(), number_children(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.18.2.14 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(), check_hepevt_consistency(), number_children(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

8.18.2.15 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().

8.18.2.16 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().

8.18.2.17 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().

8.18.2.18 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().

8.18.2.19 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().

8.18.2.20 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().

8.18.2.21 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_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

8.18.2.22 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_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

8.18.2.23 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_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

8.18.2.24 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_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

8.18.2.25 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().

8.18.2.26 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().

8.18.2.27 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().

8.18.2.28 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().

8.18.2.29 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().

8.18.2.30 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().

8.18.2.31 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().

8.18.2.32 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().

8.18.2.33 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().

8.18.2.34 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().

8.18.2.35 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().

8.18.2.36 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().

8.18.2.37 void HepMC::HEPEVT_Wrapper::set_sizeof_int (unsigned *int*) [inline, static]

define size of integer

Definition at line 232 of file HEPEVT_Wrapper.h.

8.18.2.38 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_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_PythiaParticle.cc**.

Definition at line 242 of file HEPEVT_Wrapper.h.

Referenced by main().

8.18.2.39 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_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_PythiaParticle.cc**.

Definition at line 251 of file HEPEVT_Wrapper.h.

Referenced by main().

8.18.2.40 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().

8.18.2.41 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().

8.18.2.42 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().

8.18.2.43 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.

8.18.2.44 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().

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

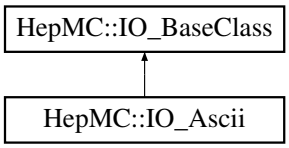
- **HEPEVT_Wrapper.h**
- **HEPEVT_Wrapper.cc**

8.19 HepMC::IO_Ascii Class Reference

IO_Ascii (p. 133) is used to read or write from an ascii file.

`#include <IO_Ascii.h>`

Inheritance diagram for HepMC::IO_Ascii::



Public Member Functions

- **IO_Ascii** (const char *filename="IO_Ascii.dat", std::ios::openmode mode=std::ios::out)
  constructor requiring a file name and std::ios mode
- virtual ~**IO_Ascii** ()
- 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. 197)*
- bool **fill_particle_data_table** (**ParticleDataTable** *)
  *fill this **ParticleDataTable** (p. 197)*
- 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

Protected Member Functions

- void **write_vertex** (**GenVertex** *)
  write vertex information
- void **write_particle** (**GenParticle** *p)
  write particle information

- void **write_particle_data** (const **ParticleData** *d)
  *write **ParticleDataTable** (p. 197) information*
- **GenVertex** * **read_vertex** (**TempParticleMap** &particle_to_end_vertex)
  read vertex information
- **GenParticle** * **read_particle** (**TempParticleMap** &particle_to_end_vertex)
  *read **GenParticle** (p. 77) information*
- **ParticleData** * **read_particle_data** (**ParticleDataTable** *)
  *read **ParticleDataTable** (p. 197) information*
- bool **write_end_listing** ()
  write end tag
- bool **search_for_key_end** (std::istream &in, const char *key)
  look for line type (key)
- bool **search_for_key_beginning** (std::istream &in, const char *key)
  not tested and NOT used anywhere!
- bool **eat_key** (std::iostream &in, const char *key)
  string manipulation accounting
- int **find_in_map** (const std::map< **GenVertex** *, int > &m, **GenVertex** *v) const
  find this vertex in the map of vertices
- void **output** (const double &)
  write double
- void **output** (const int &)
  write int
- void **output** (const long int &)
  write long int
- void **output** (const char &)
  write a single character

8.19.1 Detailed Description

IO_Ascii (p. 133) is used to read or write from an ascii file.

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_EventSelection.cc, **example_MyPythia.cc**, **example_MyPythiaRead.cc**, and **example_UsingIterators.cc**.

Definition at line 63 of file IO_Ascii.h.

8.19.2 Constructor & Destructor Documentation

8.19.2.1 HepMC::IO_Ascii::IO_Ascii (const char * *filename* = "IO_Ascii.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

Definition at line 15 of file IO_Ascii.cc.

8.19.2.2 HepMC::IO_Ascii::~IO_Ascii () [virtual]

Definition at line 34 of file IO_Ascii.cc.

References write_end_listing().

8.19.3 Member Function Documentation

8.19.3.1 void HepMC::IO_Ascii::write_event (const GenEvent * *evt*) [virtual]

write this event

Writes evt to m_file. It does NOT delete the event after writing.

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 49 of file IO_Ascii.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(), output(), HepMC::GenEvent::random_states(), HepMC::GenEvent::signal_process_id(), HepMC::GenEvent::signal_process_vertex(), HepMC::WeightContainer::size(), v, HepMC::versionName(), HepMC::GenEvent::vertices_begin(), HepMC::GenEvent::vertices_end(), HepMC::GenEvent::vertices_size(), HepMC::GenEvent::weights(), and write_vertex().

8.19.3.2 bool HepMC::IO_Ascii::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 98 of file IO_Ascii.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenEvent::add_vertex(), HepMC::GenEvent::barcode_to_vertex(), eat_key(), HepMC::TempParticleMap::end_vertex(), HepMC::TempParticleMap::order_begin(), HepMC::TempParticleMap::order_end(), p, read_vertex(), search_for_key_end(), HepMC::GenEvent::set_event_number(), HepMC::GenEvent::set_random_states(), HepMC::GenEvent::set_signal_process_id(), HepMC::GenEvent::set_signal_process_vertex(), v, and HepMC::GenEvent::weights().

8.19.3.3 void HepMC::IO_Ascii::write_particle_data_table (const ParticleDataTable *) [virtual]

write this **ParticleDataTable** (p. 197)

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 219 of file IO_Ascii.cc.

References HepMC::ParticleDataTable::begin(), HepMC::ParticleDataTable::end(), write_end_listing(), and write_particle_data().

8.19.3.4 bool HepMC::IO_Ascii::fill_particle_data_table (ParticleDataTable *) [virtual]

fill this **ParticleDataTable** (p. 197)

Implements **HepMC::IO_BaseClass** (p. 146).

Definition at line 239 of file IO_Ascii.cc.

References eat_key(), read_particle_data(), search_for_key_end(), and HepMC::ParticleDataTable::set_description().

8.19.3.5 void HepMC::IO_Ascii::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_Ascii-COMMENT\n"

Definition at line 204 of file IO_Ascii.cc.

References write_end_listing().

8.19.3.6 int HepMC::IO_Ascii::rdstate () const [inline]

check the state of the IO stream

Definition at line 140 of file IO_Ascii.h.

Referenced by main().

8.19.3.7 void HepMC::IO_Ascii::clear () [inline]

clear the IO stream

Definition at line 141 of file IO_Ascii.h.

8.19.3.8 void HepMC::IO_Ascii::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 146).

Definition at line 39 of file IO_Ascii.cc.

8.19.3.9 void HepMC::IO_Ascii::write_vertex (GenVertex *) [protected]

write vertex information

Definition at line 287 of file IO_Ascii.cc.

References output(), v, and write_particle().

Referenced by write_event().

8.19.3.10 void HepMC::IO_Ascii::write_particle (GenParticle * *p*) [protected]

write particle information

Definition at line 333 of file IO_Ascii.cc.

References output(), and p.

Referenced by write_vertex().

8.19.3.11 void HepMC::IO_Ascii::write_particle_data (const ParticleData * *d*) [protected]

write **ParticleDataTable** (p. 197) information

Definition at line 357 of file IO_Ascii.cc.

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

Referenced by write_particle_data_table().

8.19.3.12 GenVertex * HepMC::IO_Ascii::read_vertex (TempParticleMap & *particle_to_end_vertex*) [protected]

read vertex information

Definition at line 376 of file IO_Ascii.cc.

References read_particle(), and v.

Referenced by fill_next_event().

8.19.3.13 GenParticle * HepMC::IO_Ascii::read_particle (TempParticleMap & *particle_to_end_vertex*) [protected]

read **GenParticle** (p. 77) information

Definition at line 412 of file IO_Ascii.cc.

References HepMC::TempParticleMap::addEndParticle(), p, and HepMC::Flow::set_icode().

Referenced by read_vertex().

8.19.3.14 ParticleData * HepMC::IO_Ascii::read_particle_data (ParticleDataTable *) [protected]

read **ParticleDataTable** (p. 197) information

Definition at line 453 of file IO_Ascii.cc.

References HepMC::ParticleDataTable::insert().

Referenced by fill_particle_data_table().

8.19.3.15 bool HepMC::IO_Ascii::write_end_listing () [protected]

write end tag

Definition at line 475 of file IO_Ascii.cc.

Referenced by write_comment(), write_particle_data_table(), and ~IO_Ascii().

8.19.3.16 bool HepMC::IO_Ascii::search_for_key_end (std::istream & *in*, const char * *key*) [protected]

look for line type (key)

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 484 of file IO_Ascii.cc.

Referenced by fill_next_event(), fill_particle_data_table(), and search_for_key_beginning().

8.19.3.17 bool HepMC::IO_Ascii::search_for_key_beginning (std::istream & *in*, const char * *key*) [protected]

not tested and NOT used anywhere!

not tested and NOT used anywhere!

Definition at line 500 of file IO_Ascii.cc.

References search_for_key_end().

8.19.3.18 bool HepMC::IO_Ascii::eat_key (std::iostream & *in*, const char * *key*) [protected]

string manipulation accounting

eats the character string key from istream in - only if the key is the very next occurence in the stream if the key is not the next occurence, it eats nothing ... i.e. it puts back whatever it would have eaten.

Definition at line 514 of file IO_Ascii.cc.

Referenced by fill_next_event(), and fill_particle_data_table().

8.19.3.19 int HepMC::IO_Ascii::find_in_map (const std::map< GenVertex *, int > & m, GenVertex * v) const [protected]

find this vertex in the map of vertices

Definition at line 543 of file IO_Ascii.cc.

References v.

8.19.3.20 void HepMC::IO_Ascii::output (const double &) [inline, protected]

write double

Definition at line 130 of file IO_Ascii.h.

Referenced by write_event(), write_particle(), write_particle_data(), and write_vertex().

8.19.3.21 void HepMC::IO_Ascii::output (const int &) [inline, protected]

write int

Definition at line 137 of file IO_Ascii.h.

8.19.3.22 void HepMC::IO_Ascii::output (const long int &) [inline, protected]

write long int

Definition at line 138 of file IO_Ascii.h.

8.19.3.23 void HepMC::IO_Ascii::output (const char &) [inline, protected]

write a single character

Definition at line 139 of file IO_Ascii.h.

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

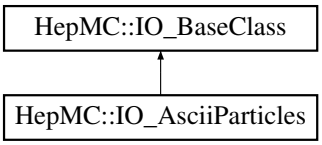
- **IO_Ascii.h**
- **IO_Ascii.cc**

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. 197)*
- bool **fill_particle_data_table** (**ParticleDataTable** *)
  *fill this **ParticleDataTable** (p. 197)*
- 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_PythiaParticle.cc.

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::write_event (const GenEvent * *evt*) [virtual]

write this event

Implements **HepMC::IO_BaseClass** (p. 145).

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::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().

8.20.3.2 bool HepMC::IO_AsciiParticles::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 180 of file IO_AsciiParticles.cc.

8.20.3.3 void HepMC::IO_AsciiParticles::write_particle_data_table (const ParticleDataTable *) [inline, virtual]

write this **ParticleDataTable** (p. 197)

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 106 of file IO_AsciiParticles.h.

8.20.3.4 bool HepMC::IO_AsciiParticles::fill_particle_data_table (ParticleDataTable *) [inline, virtual]

fill this **ParticleDataTable** (p. 197)

Implements **HepMC::IO_BaseClass** (p. 146).

Definition at line 107 of file IO_AsciiParticles.h.

8.20.3.5 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().

8.20.3.6 void HepMC::IO_AsciiParticles::setPrecision (int *iprec*) [inline]

set output precision

Definition at line 100 of file IO_AsciiParticles.h.

8.20.3.7 int HepMC::IO_AsciiParticles::rdstate () const [inline]

check the state of the IO stream

Definition at line 98 of file IO_AsciiParticles.h.

8.20.3.8 void HepMC::IO_AsciiParticles::clear () [inline]

clear the IO stream

Definition at line 99 of file IO_AsciiParticles.h.

8.20.3.9 void HepMC::IO_AsciiParticles::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 146).

Definition at line 54 of file IO_AsciiParticles.cc.

8.20.3.10 bool HepMC::IO_AsciiParticles::write_end_listing () [protected]

write end tag

Definition at line 218 of file IO_AsciiParticles.cc.

Referenced by write_comment().

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

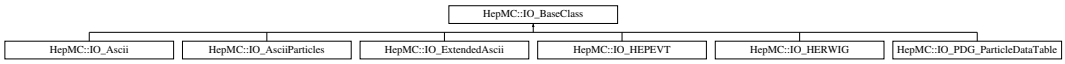
- **IO_AsciiParticles.h**
- **IO_AsciiParticles.cc**

8.21 HepMC::IO_BaseClass Class Reference

all input/output classes inherit from **IO_BaseClass** (p. 144)

`#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. 46)*
- virtual bool **fill_next_event** (**GenEvent** *)=0
  *fill this **GenEvent** (p. 46)*
- virtual void **write_particle_data_table** (const **ParticleDataTable** *)=0
  *write this **ParticleDataTable** (p. 197)*
- virtual bool **fill_particle_data_table** (**ParticleDataTable** *)=0
  *fill this **ParticleDataTable** (p. 197)*
- 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. 144)

If you want to write a new IO class, then inherit from this class and re-define read_event() and **write_event()** (p. 145)

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 void HepMC::IO_BaseClass::write_event (const GenEvent *) [pure virtual]

write this **GenEvent** (p. 46)

Implemented in **HepMC::IO_Ascii** (p. 135), **HepMC::IO_AsciiParticles** (p. 141), **HepMC::IO_ExtendedAscii** (p. 150), and **HepMC::IO_HEPEVT** (p. 159).

Referenced by operator<<().

8.21.3.2 virtual bool HepMC::IO_BaseClass::fill_next_event (GenEvent *) [pure virtual]

fill this **GenEvent** (p. 46)

Implemented in **HepMC::IO_Ascii** (p. 135), **HepMC::IO_AsciiParticles** (p. 141), **HepMC::IO_ExtendedAscii** (p. 151), **HepMC::IO_HEPEVT** (p. 158), and **HepMC::IO_HERWIG** (p. 164).

Referenced by read_next_event().

8.21.3.3 virtual void HepMC::IO_BaseClass::write_particle_data_table (const ParticleDataTable *) [pure virtual]

write this **ParticleDataTable** (p. 197)

Implemented in **HepMC::IO_Ascii** (p. 136), **HepMC::IO_AsciiParticles** (p. 142), and **HepMC::IO_ExtendedAscii** (p. 151).

Referenced by operator<<().

8.21.3.4 virtual bool HepMC::IO_BaseClass::fill_particle_data_table (ParticleDataTable *) [pure virtual]

fill this **ParticleDataTable** (p. 197)

Implemented in **HepMC::IO_Ascii** (p. 136), **HepMC::IO_AsciiParticles** (p. 142), **HepMC::IO_ExtendedAscii** (p. 151), and **HepMC::IO_PDG_ParticleDataTable** (p. 170).

Referenced by read_particle_data_table().

8.21.3.5 void HepMC::IO_BaseClass::print (std::ostream & *ostr* = std::cout) const [inline, virtual]

write output to ostr

Reimplemented in **HepMC::IO_Ascii** (p. 136), **HepMC::IO_AsciiParticles** (p. 142), **HepMC::IO_ExtendedAscii** (p. 152), **HepMC::IO_HEPEVT** (p. 159), **HepMC::IO_HERWIG** (p. 164), and **HepMC::IO_PDG_ParticleDataTable** (p. 170).

Definition at line 117 of file IO_BaseClass.h.

8.21.3.6 GenEvent * HepMC::IO_BaseClass::read_next_event () [inline]

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_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_PythiaParticle.cc**.

Definition at line 87 of file IO_BaseClass.h.

References fill_next_event().

Referenced by main(), and operator>>().

8.21.3.7 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 main(), and operator>>().

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 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().

8.21.3.10 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().

8.21.3.11 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().

8.21.3.12 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().

8.21.3.13 ParticleDataTable *& HepMC::IO_BaseClass::operator<< (ParticleDataTable *&) [inline, virtual]

the same as write_particle_data_table

Definition at line 149 of file IO_BaseClass.h.

References write_particle_data_table().

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

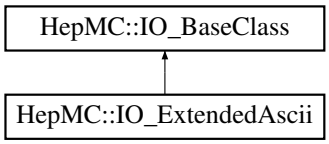
- **IO_BaseClass.h**

8.22 HepMC::IO_ExtendedAscii Class Reference

IO_ExtendedAscii (p. 148) also deals with **HeavyIon** (p. 112) and **PdfInfo** (p. 204).

`#include <IO_ExtendedAscii.h>`

Inheritance diagram for HepMC::IO_ExtendedAscii::



Public Member Functions

- **IO_ExtendedAscii** (const char *filename="IO_ExtendedAscii.dat", std::ios::openmode mode=std::ios::out)

  constructor requiring a file name and std::ios mode

- virtual ~**IO_ExtendedAscii** ()
- 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. 197)*

- bool **fill_particle_data_table** (**ParticleDataTable** *)

  *fill this **ParticleDataTable** (p. 197)*

- 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

Protected Member Functions

- void **write_vertex** (**GenVertex** *)

  write vertex information

- void **write_beam_particles** (std::pair< **HepMC::GenParticle** *, **HepMC::GenParticle** * >)

  write beam particle information

- void **write_heavy_ion** (**HeavyIon** *)

  write heavy ion information

- void **write_pdf_info** (**PdfInfo** *)

  write PDF information

- void **write_particle** (**GenParticle** *p)

  write particle information

- void **write_particle_data** (const **ParticleData** *d)

  write particle data information

- **GenVertex** * **read_vertex** (**TempParticleMap** &particle_to_end_vertex)

  read vertex information

- **GenParticle** * **read_particle** (**TempParticleMap** &particle_to_end_vertex)

  *read **GenParticle** (p. 77) information*

- **ParticleData** * **read_particle_data** (**ParticleDataTable** *)

  read particle data table information

- **HeavyIon** * **read_heavy_ion** ()

  read heavy ion information

- **PdfInfo** * **read_pdf_info** ()

  read PDF information

- bool **write_end_listing** ()

  write end tag

- bool **search_for_key_end** (std::istream &in, const char *key)

  look for line type (key)

- bool **search_for_key_beginning** (std::istream &in, const char *key)

  look for line type (key)

- bool **eat_key** (std::iostream &in, const char *key)

  string manipulation accounting

- int **find_in_map** (const std::map< **HepMC::GenVertex** *, int > &m, **GenVertex** *v) const

  find this vertex in the map of vertices

- void **output** (const double &)

  write double

- void **output** (const float &)
  write float
- void **output** (const int &)
  write int
- void **output** (const long int &)
  write long int
- void **output** (const char &)
  write a single character

8.22.1 Detailed Description

IO_ExtendedAscii (p. 148) also deals with **HeavyIon** (p. 112) and **PdfInfo** (p. 204).

event input/output in ascii format for machine reading extended format contains **HeavyIon** (p. 112) and **PdfInfo** (p. 204) classes

Examples:

example_MyPythia.cc.

Definition at line 64 of file IO_ExtendedAscii.h.

8.22.2 Constructor & Destructor Documentation

8.22.2.1 HepMC::IO_ExtendedAscii::IO_ExtendedAscii (const char * *filename* = "IO_ExtendedAscii.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

Definition at line 18 of file IO_ExtendedAscii.cc.

8.22.2.2 HepMC::IO_ExtendedAscii::~IO_ExtendedAscii () [virtual]

Definition at line 37 of file IO_ExtendedAscii.cc.

References write_end_listing().

8.22.3 Member Function Documentation

8.22.3.1 void HepMC::IO_ExtendedAscii::write_event (const GenEvent * *evt*) [virtual]

write this event

Writes evt to m_file. It does NOT delete the event after writing.

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 52 of file IO_ExtendedAscii.cc.

References HepMC::GenEvent::alphaQCD(), HepMC::GenEvent::alphaQED(), HepMC::GenVertex::barcode(), HepMC::GenEvent::beam_particles(), HepMC::WeightContainer::begin(), HepMC::WeightContainer::end(), HepMC::GenEvent::event_number(), HepMC::GenEvent::event_scale(), HepMC::GenEvent::heavy_ion(), HepMC::GenEvent::mpi(), output(), HepMC::GenEvent::pdf_info(), HepMC::GenEvent::random_states(), HepMC::GenEvent::signal_process_id(), HepMC::GenEvent::signal_process_vertex(), HepMC::WeightContainer::size(), v, HepMC::versionName(), HepMC::GenEvent::vertices_begin(), HepMC::GenEvent::vertices_end(), HepMC::GenEvent::vertices_size(), HepMC::GenEvent::weights(), write_beam_particles(), write_heavy_ion(), write_pdf_info(), and write_vertex().

8.22.3.2 bool HepMC::IO_ExtendedAscii::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 105 of file IO_ExtendedAscii.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenEvent::add_vertex(), HepMC::GenEvent::barcode_to_vertex(), eat_key(), HepMC::TempParticleMap::end_vertex(), HepMC::TempParticleMap::order_begin(), HepMC::TempParticleMap::order_end(), p, read_heavy_ion(), read_pdf_info(), read_vertex(), search_for_key_end(), HepMC::GenEvent::set_beam_particles(), HepMC::GenEvent::set_event_number(), HepMC::GenEvent::set_heavy_ion(), HepMC::GenEvent::set_mpi(), HepMC::GenEvent::set_pdf_info(), HepMC::GenEvent::set_random_states(), HepMC::GenEvent::set_signal_process_id(), HepMC::GenEvent::set_signal_process_vertex(), v, and HepMC::GenEvent::weights().

8.22.3.3 void HepMC::IO_ExtendedAscii::write_particle_data_table (const ParticleDataTable *) [virtual]

write this **ParticleDataTable** (p. 197)

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 239 of file IO_ExtendedAscii.cc.

References HepMC::ParticleDataTable::begin(), HepMC::ParticleDataTable::end(), write_end_listing(), and write_particle_data().

8.22.3.4 bool HepMC::IO_ExtendedAscii::fill_particle_data_table (ParticleDataTable *) [virtual]

fill this **ParticleDataTable** (p. 197)

Implements **HepMC::IO_BaseClass** (p. 146).

Definition at line 259 of file IO_ExtendedAscii.cc.

References eat_key(), read_particle_data(), search_for_key_end(), and HepMC::ParticleDataTable::set_description().

8.22.3.5 void HepMC::IO_ExtendedAscii::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_ExtendedAscii-COMMENT\n"

Definition at line 224 of file IO_ExtendedAscii.cc.

References write_end_listing().

8.22.3.6 int HepMC::IO_ExtendedAscii::rdstate () const [inline]

check the state of the IO stream

Definition at line 159 of file IO_ExtendedAscii.h.

8.22.3.7 void HepMC::IO_ExtendedAscii::clear () [inline]

clear the IO stream

Definition at line 160 of file IO_ExtendedAscii.h.

8.22.3.8 void HepMC::IO_ExtendedAscii::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 146).

Definition at line 42 of file IO_ExtendedAscii.cc.

8.22.3.9 void HepMC::IO_ExtendedAscii::write_vertex (GenVertex *) [protected]

write vertex information

Definition at line 307 of file IO_ExtendedAscii.cc.

References output(), v, and write_particle().

Referenced by write_event().

8.22.3.10 void HepMC::IO_ExtendedAscii::write_beam_particles (std::pair< HepMC::GenParticle *, HepMC::GenParticle * >) [protected]

write beam particle information

Referenced by write_event().

8.22.3.11 void HepMC::IO_ExtendedAscii::write_heavy_ion (HeavyIon *) [protected]

write heavy ion information

Definition at line 370 of file IO_ExtendedAscii.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(), output(), HepMC::HeavyIon::sigma_inel_NN(), HepMC::HeavyIon::spectator_neutrons(), and HepMC::HeavyIon::spectator_protons().

Referenced by write_event().

8.22.3.12 void HepMC::IO_ExtendedAscii::write_pdf_info (PdfInfo *) [protected]

write PDF information

Definition at line 414 of file IO_ExtendedAscii.cc.

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

Referenced by write_event().

8.22.3.13 void HepMC::IO_ExtendedAscii::write_particle (GenParticle * *p*) [protected]

write particle information

Definition at line 446 of file IO_ExtendedAscii.cc.

References output(), and p.

Referenced by write_vertex().

8.22.3.14 void HepMC::IO_ExtendedAscii::write_particle_data (const ParticleData * *d*) [protected]

write particle data information

Definition at line 471 of file IO_ExtendedAscii.cc.

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

Referenced by write_particle_data_table().

8.22.3.15 GenVertex * HepMC::IO_ExtendedAscii::read_vertex (TempParticleMap & *particle_to_end_vertex*) [protected]

read vertex information

Definition at line 490 of file IO_ExtendedAscii.cc.

References read_particle(), and v.

Referenced by fill_next_event().

8.22.3.16 GenParticle * HepMC::IO_ExtendedAscii::read_particle (TempParticleMap & *particle_to_end_vertex*) [protected]

read **GenParticle** (p. 77) information

Definition at line 575 of file IO_ExtendedAscii.cc.

References HepMC::TempParticleMap::addEndParticle(), p, and HepMC::Flow::set_icode().

Referenced by read_vertex().

8.22.3.17 ParticleData * HepMC::IO_ExtendedAscii::read_particle_data (ParticleDataTable *) [protected]

read particle data table information

Definition at line 617 of file IO_ExtendedAscii.cc.

References HepMC::ParticleDataTable::insert().

Referenced by fill_particle_data_table().

8.22.3.18 HeavyIon * HepMC::IO_ExtendedAscii::read_heavy_ion () [protected]

read heavy ion information

Definition at line 526 of file IO_ExtendedAscii.cc.

Referenced by fill_next_event().

8.22.3.19 PdfInfo * HepMC::IO_ExtendedAscii::read_pdf_info () [protected]

read PDF information

Definition at line 552 of file IO_ExtendedAscii.cc.

Referenced by fill_next_event().

8.22.3.20 bool HepMC::IO_ExtendedAscii::write_end_listing () [protected]

write end tag

Definition at line 639 of file IO_ExtendedAscii.cc.

Referenced by write_comment(), write_particle_data_table(), and ~IO_ExtendedAscii().

8.22.3.21 bool HepMC::IO_ExtendedAscii::search_for_key_end (std::istream & *in*, const char * *key*) [protected]

look for line type (key)

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 648 of file IO_ExtendedAscii.cc.

Referenced by fill_next_event(), fill_particle_data_table(), and search_for_key_beginning().

8.22.3.22 bool HepMC::IO_ExtendedAscii::search_for_key_beginning (std::istream & *in*, const char * *key*) [protected]

look for line type (key)

not tested and NOT used anywhere!

Definition at line 664 of file IO_ExtendedAscii.cc.

References search_for_key_end().

8.22.3.23 bool HepMC::IO_ExtendedAscii::eat_key (std::iostream & *in*, const char * *key*) [protected]

string manipulation accounting

eats the character string key from istream in - only if the key is the very next occurence in the stream if the key is not the next occurence, it eats nothing ... i.e. it puts back whatever it would have eaten.

Definition at line 678 of file IO_ExtendedAscii.cc.

Referenced by fill_next_event(), and fill_particle_data_table().

8.22.3.24 int HepMC::IO_ExtendedAscii::find_in_map (const std::map< HepMC::GenVertex *, int > & *m*, GenVertex * *v*) const [protected]

find this vertex in the map of vertices

8.22.3.25 void HepMC::IO_ExtendedAscii::output (const double &) [inline, protected]

write double

Definition at line 142 of file IO_ExtendedAscii.h.

Referenced by write_event(), write_heavy_ion(), write_particle(), write_particle_data(), write_pdf_info(), and write_vertex().

8.22.3.26 void HepMC::IO_ExtendedAscii::output (const float &) [inline, protected]

write float

Definition at line 149 of file IO_ExtendedAscii.h.

8.22.3.27 void HepMC::IO_ExtendedAscii::output (const int &) [inline, protected]

write int

Definition at line 156 of file IO_ExtendedAscii.h.

8.22.3.28 void HepMC::IO_ExtendedAscii::output (const long int &) [inline, protected]

write long int

Definition at line 157 of file IO_ExtendedAscii.h.

8.22.3.29 void HepMC::IO_ExtendedAscii::output (const char &) [inline, protected]

write a single character

Definition at line 158 of file IO_ExtendedAscii.h.

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

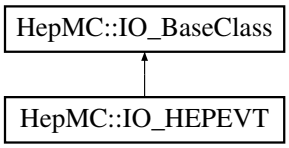
- **IO_ExtendedAscii.h**
- **IO_ExtendedAscii.cc**

8.23 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. 46)*
- void **write_event** (const **GenEvent** *)
  *write this **GenEvent** (p. 46)*
- 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
- void **set_trust_mothers_before_daughters** (bool b=1)
  define mother daughter trust rules
- void **set_trust_both_mothers_and_daughters** (bool b=0)
  define mother daughter trust rules
- void **set_print_inconsistency_errors** (bool b=1)

Protected Member Functions

- **GenParticle** * **build_particle** (int index)
  *create a **GenParticle** (p. 77)*

- 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

8.23.1 Detailed Description

HEPEVT IO class.

IO class for reading the standard HEPEVT common block.

Examples:

example_MyPythia.cc, **example_MyPythiaOnlyToHepMC.cc**, **example_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_PythiaParticle.cc**.

Definition at line 40 of file IO_HEPEVT.h.

8.23.2 Constructor & Destructor Documentation

8.23.2.1 HepMC::IO_HEPEVT::IO_HEPEVT ()

Definition at line 12 of file IO_HEPEVT.cc.

8.23.2.2 HepMC::IO_HEPEVT::~IO_HEPEVT () [virtual]

Definition at line 17 of file IO_HEPEVT.cc.

8.23.3 Member Function Documentation

8.23.3.1 bool HepMC::IO_HEPEVT::fill_next_event (GenEvent *) [virtual]

fill this **GenEvent** (p. 46)

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 30 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(), and HepMC::GenEvent::set_event_number().

8.23.3.2 void HepMC::IO_HEPEVT::write_event (const GenEvent *) [virtual]

write this **GenEvent** (p. 46)

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 107 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().

8.23.3.3 void HepMC::IO_HEPEVT::print (std::ostream & *ostr* = std::cout) const [virtual]

write output to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 146).

Definition at line 19 of file IO_HEPEVT.cc.

8.23.3.4 bool HepMC::IO_HEPEVT::trust_both_mothers_and_daughters () const [inline]

default is false

Definition at line 115 of file IO_HEPEVT.h.

8.23.3.5 bool HepMC::IO_HEPEVT::trust_mothers_before_daughters () const [inline]

default is true

Definition at line 118 of file IO_HEPEVT.h.

8.23.3.6 bool HepMC::IO_HEPEVT::print_inconsistency_errors () const [inline]

default is true

Definition at line 121 of file IO_HEPEVT.h.

8.23.3.7 void HepMC::IO_HEPEVT::set_trust_mothers_before_daughters (bool *b* = 1) [inline]

define mother daughter trust rules

Definition at line 127 of file IO_HEPEVT.h.

8.23.3.8 void HepMC::IO_HEPEVT::set_trust_both_mothers_and_daughters (bool *b* = 0) [inline]

define mother daughter trust rules

Definition at line 124 of file IO_HEPEVT.h.

8.23.3.9 void HepMC::IO_HEPEVT::set_print_inconsistency_errors (bool *b* = 1) [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. 123)

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 130 of file IO_HEPEVT.h.

8.23.3.10 GenParticle * HepMC::IO_HEPEVT::build_particle (int *index*) [protected]

create a **GenParticle** (p. 77)

Builds a particle object corresponding to index in HEPEVT

Definition at line 319 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().

8.23.3.11 void HepMC::IO_HEPEVT::build_production_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create a production vertex

Referenced by fill_next_event().

8.23.3.12 void HepMC::IO_HEPEVT::build_end_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create an end vertex

Referenced by fill_next_event().

8.23.3.13 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

Referenced by write_event().

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

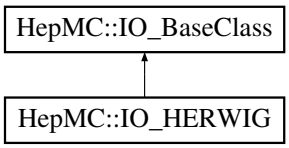
- **IO_HEPEVT.h**
- **IO_HEPEVT.cc**

8.24 HepMC::IO_HERWIG Class Reference

IO_HERWIG (p. 162) 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=1)
  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=1)
  define mother daughter trust rules
- void **set_trust_both_mothers_and_daughters** (bool b=0)

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

8.24.1 Detailed Description

IO_HERWIG (p. 162) is used to get Herwig information.

IO class for reading the HEPEVT common block from the Herwig monte carlo program.

Examples:

example_MyHerwig.cc.

Definition at line 57 of file IO_HERWIG.h.

8.24.2 Constructor & Destructor Documentation

8.24.2.1 HepMC::IO_HERWIG::IO_HERWIG ()

Definition at line 12 of file IO_HERWIG.cc.

8.24.2.2 HepMC::IO_HERWIG::~IO_HERWIG () [virtual]

Definition at line 81 of file IO_HERWIG.cc.

8.24.3 Member Function Documentation

8.24.3.1 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. 46) return T/F =success/failure

sufficient to do one or the other.

Implements **HepMC::IO_BaseClass** (p. 145).

Definition at line 94 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::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().

8.24.3.2 void HepMC::IO_HERWIG::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 146).

Definition at line 83 of file IO_HERWIG.cc.

8.24.3.3 double HepMC::IO_HERWIG::interfaces_to_version_number () const [inline]

this information is dubious

Definition at line 66 of file IO_HERWIG.h.

8.24.3.4 bool HepMC::IO_HERWIG::print_inconsistency_errors () const [inline]

default is true

Definition at line 149 of file IO_HERWIG.h.

8.24.3.5 void HepMC::IO_HERWIG::set_print_inconsistency_errors (bool *b* = 1) [inline]

decide whether or not to print inconsistency errors

Definition at line 158 of file IO_HERWIG.h.

8.24.3.6 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.

8.24.3.7 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. 162). 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.

8.24.3.8 bool HepMC::IO_HERWIG::trust_both_mothers_and_daughters () const [inline, protected]

default is true

Definition at line 143 of file IO_HERWIG.h.

8.24.3.9 bool HepMC::IO_HERWIG::trust_mothers_before_daughters () const [inline, protected]

default is false

Definition at line 146 of file IO_HERWIG.h.

8.24.3.10 void HepMC::IO_HERWIG::set_trust_mothers_before_daughters (bool *b* = 1) [inline, protected]

define mother daughter trust rules

Definition at line 155 of file IO_HERWIG.h.

8.24.3.11 void HepMC::IO_HERWIG::set_trust_both_mothers_and_daughters (bool *b* = 0) [inline, protected]

define mother daughter trust rules

Definition at line 152 of file IO_HERWIG.h.

8.24.3.12 GenParticle * HepMC::IO_HERWIG::build_particle (int *index*) [protected]

make a particle

Builds a particle object corresponding to index in HEPEVT

Definition at line 342 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().

8.24.3.13 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 201 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 build_end_vertex(), and fill_next_event().

8.24.3.14 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 274 of file IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), build_production_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().

8.24.3.15 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 357 of file IO_HERWIG.cc.

References p.

8.24.3.16 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 subrpocess particles 100 (ID=0 cone)

Special HERWIG particle id's 91 clusters 94 jets 0 others with no pdg code

Definition at line 364 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().

8.24.3.17 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 zeero-ed entries as we go and making sure we keep the mother/daughter relationships appropriate

Definition at line 637 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().

8.24.3.18 void HepMC::IO_HERWIG::zero_hepevt_entry (int *i*) const [protected]

zero out a HEPEVT pseudo particle

Definition at line 697 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().

8.24.3.19 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 708 of file IO_HERWIG.cc.

Referenced by repair_hepevt().

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

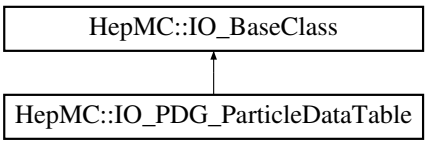
- **IO_HERWIG.h**
- **IO_HERWIG.cc**

8.25 HepMC::IO_PDG_ParticleDataTable Class Reference

an example **ParticleDataTable** (p. 197) 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

8.25.1 Detailed Description

an example **ParticleDataTable** (p. 197) IO method

Example of reading from file PDG98_ParticleDataTable.txt

Definition at line 49 of file IO_PDG_ParticleDataTable.h.

8.25.2 Constructor & Destructor Documentation

8.25.2.1 HepMC::IO_PDG_ParticleDataTable::IO_PDG_ParticleDataTable (const char * *filename* = "PDG98_ParticleDataTable.txt")

constructor using filename

Definition at line 17 of file IO_PDG_ParticleDataTable.cc.

8.25.2.2 HepMC::IO_PDG_ParticleDataTable::~IO_PDG_ParticleDataTable () [virtual]

Definition at line 20 of file IO_PDG_ParticleDataTable.cc.

8.25.3 Member Function Documentation

8.25.3.1 bool HepMC::IO_PDG_ParticleDataTable::fill_particle_data_table (ParticleDataTable *) [virtual]

read the input and fill the table

Implements **HepMC::IO_BaseClass** (p. 146).

Definition at line 24 of file IO_PDG_ParticleDataTable.cc.

References read_entry(), search_for_key_end(), and HepMC::ParticleDataTable::set_description().

8.25.3.2 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 165 of file IO_PDG_ParticleDataTable.cc.

References HepMC::ParticleDataTable::erase(), HepMC::ParticleDataTable::find(), HepMC::ParticleDataTable::insert(), and HepMC::ParticleData::mass().

Referenced by main().

8.25.3.3 void HepMC::IO_PDG_ParticleDataTable::print (std::ostream & *ostr* = std::cout) const [inline, virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 146).

Definition at line 85 of file IO_PDG_ParticleDataTable.h.

8.25.3.4 int HepMC::IO_PDG_ParticleDataTable::rdstate () const [inline]

check the IO state

Definition at line 63 of file IO_PDG_ParticleDataTable.h.

Referenced by main().

8.25.3.5 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** (p. 133) 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 203 of file IO_PDG_ParticleDataTable.cc.

Referenced by fill_particle_data_table().

8.25.3.6 void HepMC::IO_PDG_ParticleDataTable::read_entry (ParticleDataTable *) [protected]

read a line

Definition at line 63 of file IO_PDG_ParticleDataTable.cc.

References HepMC::clifetime_from_width(), HepMC::ParticleDataTable::find(), HepMC::ParticleDataTable::insert(), HepMC::ParticleData::set_clifetime(), and HepMC::ParticleData::set_mass().

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**

8.26 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

8.26.1 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.

8.26.2 Member Data Documentation

8.26.2.1 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**

8.27 HepMC::detail::is_arithmetic< char > Struct Template Reference

character is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

8.27.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< char >

character is arithmetic

Definition at line 29 of file is_arithmetic.h.

8.27.2 Member Data Documentation

8.27.2.1 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**

8.28 HepMC::detail::is_arithmetic< double > Struct Template Reference

double 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< double >

double is arithmetic

Definition at line 79 of file is_arithmetic.h.

8.28.2 Member Data Documentation

8.28.2.1 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**

8.29 HepMC::detail::is_arithmetic< float > Struct Template Reference

float 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< float >

float is arithmetic

Definition at line 74 of file is_arithmetic.h.

8.29.2 Member Data Documentation

8.29.2.1 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**

8.30 HepMC::detail::is_arithmetic< int > Struct Template Reference

int 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< int >

int is arithmetic

Definition at line 54 of file is_arithmetic.h.

8.30.2 Member Data Documentation

8.30.2.1 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**

8.31 HepMC::detail::is_arithmetic< long > Struct Template Reference

long 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< long >

long is arithmetic

Definition at line 64 of file is_arithmetic.h.

8.31.2 Member Data Documentation

8.31.2.1 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**

8.32 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

8.32.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< long double >

long double is arithmetic

Definition at line 84 of file is_arithmetic.h.

8.32.2 Member Data Documentation

8.32.2.1 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**

8.33 HepMC::detail::is_arithmetic< short > Struct Template Reference

short 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< short >

short is arithmetic

Definition at line 44 of file is_arithmetic.h.

8.33.2 Member Data Documentation

8.33.2.1 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**

8.34 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

8.34.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< signed char >

signed character is arithmetic

Definition at line 39 of file is_arithmetic.h.

8.34.2 Member Data Documentation

8.34.2.1 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**

8.35 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

8.35.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned char >

unsigned character is arithmetic

Definition at line 34 of file is_arithmetic.h.

8.35.2 Member Data Documentation

8.35.2.1 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**

8.36 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

8.36.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned int >

unsigned int is arithmetic

Definition at line 59 of file is_arithmetic.h.

8.36.2 Member Data Documentation

8.36.2.1 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**

8.37 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

8.37.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned long >

unsigned long is arithmetic

Definition at line 69 of file is_arithmetic.h.

8.37.2 Member Data Documentation

8.37.2.1 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**

8.38 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

8.38.1 Detailed Description

template<> struct HepMC::detail::is_arithmetic< unsigned short >

unsigned short is arithmetic

Definition at line 49 of file is_arithmetic.h.

8.38.2 Member Data Documentation

8.38.2.1 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**

8.39 IsFinalState Class Reference

example class

`#include <testHepMCIteration.h>`

Public Member Functions

- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle does not decay
- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle does not decay

8.39.1 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.

8.39.2 Member Function Documentation

8.39.2.1 bool IsFinalState::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.

8.39.2.2 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 files:

- **example_UsingIterators.cc**
- **testHepMCIteration.h**

8.40 IsGoodEvent Class Reference

example class

`#include <IsGoodEvent.h>`

Public Member Functions

- bool **operator()** (const **HepMC::GenEvent** *evt)
  check this event for goodness
- bool **operator()** (const **HepMC::GenEvent** *evt)

8.40.1 Detailed Description

example class

event selection predicate. returns true if the event contains a photon with pT > 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 IsGoodEvent::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().

8.40.2.2 bool IsGoodEvent::operator() (const HepMC::GenEvent * *evt*) [inline]

Definition at line 14 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 files:

- **example_EventSelection.cc**
- **IsGoodEvent.h**

8.41 IsGoodEventMyPythia Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenEvent** *evt)
  returns true if event is "good"

8.41.1 Detailed Description

example class

event selection predicate. returns true if the event contains a photon with pT $>$ 25 GeV

Examples:

example_MyPythiaWithEventSelection.cc.

Definition at line 28 of file example_MyPythiaWithEventSelection.cc.

8.41.2 Member Function Documentation

8.41.2.1 bool IsGoodEventMyPythia::operator() (const HepMC::GenEvent * *evt*) [inline]

returns true if event is "good"

Examples:

example_MyPythiaWithEventSelection.cc.

Definition at line 31 of file example_MyPythiaWithEventSelection.cc.

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

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

- **example_MyPythiaWithEventSelection.cc**

8.42 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

8.42.1 Detailed Description

example class

this predicate returns true if the input particle is a photon in the central region (eta < 2.5) with pT > 10 GeV

Examples:

example_UsingIterators.cc.

Definition at line 20 of file example_UsingIterators.cc.

8.42.2 Member Function Documentation

8.42.2.1 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**

8.43 IsW_Boson Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenParticle** *p)
  returns true if the GenParticle is a W

8.43.1 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.

8.43.2 Member Function Documentation

8.43.2.1 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**

8.44 HepMC::ParticleData Class Reference

an example **ParticleData** (p. 190) 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. 190) *objects in memory*

Friends

- std::ostream & **operator<<** (std::ostream &, const **ParticleData** &)
  write to ostr

8.44.1 Detailed Description

an example **ParticleData** (p. 190) class

Particle Data common to all particles of a given PDG id

Examples:

example_BuildEventFromScratch.cc.

Definition at line 69 of file ParticleData.h.

8.44.2 Constructor & Destructor Documentation

8.44.2.1 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 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().

8.44.2.2 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. 190)(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().

8.44.2.3 HepMC::ParticleData::~ParticleData () [virtual]

Definition at line 44 of file ParticleData.cc.

8.44.3 Member Function Documentation

8.44.3.1 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.

8.44.3.2 bool HepMC::ParticleData::operator!= (const ParticleData &) const [inline]

inequality

Definition at line 222 of file ParticleData.h.

References pdg_id().

8.44.3.3 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().

8.44.3.4 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().

8.44.3.5 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().

8.44.3.6 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().

8.44.3.7 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().

8.44.3.8 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().

8.44.3.9 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().

8.44.3.10 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<<(), print(), HepMC::IO_ExtendedAscii::write_particle_data(), and HepMC::IO_Ascii::write_particle_data().

8.44.3.11 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<<(), print(), HepMC::IO_ExtendedAscii::write_particle_data(), and HepMC::IO_Ascii::write_particle_data().

8.44.3.12 double HepMC::ParticleData::charge () const [inline]

charge

Definition at line 175 of file ParticleData.h.

Referenced by HepMC::operator<<(), print(), HepMC::IO_ExtendedAscii::write_particle_data(), and HepMC::IO_Ascii::write_particle_data().

8.44.3.13 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<<(), print(), HepMC::IO_ExtendedAscii::write_particle_data(), and HepMC::IO_Ascii::write_particle_data().

8.44.3.14 double HepMC::ParticleData::width () const

width as calculated from clifetime

Definition at line 71 of file ParticleData.cc.

References HepMC::HepMC_hbarc.

8.44.3.15 double HepMC::ParticleData::clifetime () const [inline]

lifetime in mm

Definition at line 179 of file ParticleData.h.

Referenced by HepMC::operator<<(), print(), HepMC::IO_ExtendedAscii::write_particle_data(), and HepMC::IO_Ascii::write_particle_data().

8.44.3.16 double HepMC::ParticleData::spin () const [inline]

J spin.

Definition at line 180 of file ParticleData.h.

Referenced by HepMC::operator<<(), print(), HepMC::IO_ExtendedAscii::write_particle_data(), and HepMC::IO_Ascii::write_particle_data().

8.44.3.17 void HepMC::ParticleData::set_charge (double) [inline]

set charge

Definition at line 181 of file ParticleData.h.

Referenced by ParticleData().

8.44.3.18 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().

8.44.3.19 void HepMC::ParticleData::set_width (double) [inline]

set width

Definition at line 193 of file ParticleData.h.

References HepMC::HepMC_hbarc.

8.44.3.20 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().

8.44.3.21 void HepMC::ParticleData::set_spin (double) [inline]

set J spin

Definition at line 205 of file ParticleData.h.

Referenced by ParticleData().

8.44.3.22 unsigned int HepMC::ParticleData::counter () [static, protected]

num **ParticleData** (p. 190) objects in memory

Definition at line 86 of file ParticleData.cc.

8.44.3.23 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 en excited electron (40000011) would be returned as 11 Useful only internally for sorting particles!

Definition at line 61 of file ParticleData.cc.

8.44.4 Friends And Related Function Documentation

8.44.4.1 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**

8.45 HepMC::ParticleDataTable Class Reference

an example **ParticleDataTable** (p. 197) class

`#include <ParticleDataTable.h>`

Public Types

- typedef std::map< int, **HepMC::ParticleData** * >::**iterator iterator**
  *iterator for **ParticleData*** (p. *190*) *map*
- typedef std::map< int, **HepMC::ParticleData** * >::**const_iterator const_iterator**
  *const iterator for **ParticleData*** (p. *190*) *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. *190*) *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. *190*) *instances in this table*
- void **clear** ()
  clears table without deleting
- **ParticleData** * **operator[]** (int id) const
  *return pointer to requested **ParticleData*** (p. *190*)

- **ParticleData** * **find** (int id) const
  return pointer to requested ***ParticleData*** *(*p. *190)*
- 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

8.45.1 Detailed Description

an example **ParticleDataTable** (p. 197) class

Example container for **ParticleData** (p. 190) instances. Basically just an interface to STL map.

Examples:

example_BuildEventFromScratch.cc.

Definition at line 35 of file ParticleDataTable.h.

8.45.2 Member Typedef Documentation

8.45.2.1 typedef std::map<int,HepMC::ParticleData*>::iterator HepMC::ParticleDataTable::iterator

iterator for **ParticleData** (p. 190) map

Definition at line 75 of file ParticleDataTable.h.

8.45.2.2 typedef std::map<int,HepMC::ParticleData*>::const_iterator HepMC::ParticleDataTable::const_iterator

const iterator for **ParticleData** (p. 190) map

Definition at line 77 of file ParticleDataTable.h.

8.45.3 Constructor & Destructor Documentation

8.45.3.1 HepMC::ParticleDataTable::ParticleDataTable (std::string *description* = std::string()) [inline]

constructor with optional description

Definition at line 107 of file ParticleDataTable.h.

8.45.3.2 HepMC::ParticleDataTable::ParticleDataTable (const char *description*) [inline]

constructor with description

Definition at line 110 of file ParticleDataTable.h.

8.45.3.3 HepMC::ParticleDataTable::ParticleDataTable (const ParticleDataTable &) [inline]

copy constructor

Definition at line 114 of file ParticleDataTable.h.

8.45.3.4 HepMC::ParticleDataTable::~ParticleDataTable () [inline, virtual]

Shallow: does not delete **ParticleData** (p. 190) entries.

Definition at line 118 of file ParticleDataTable.h.

8.45.4 Member Function Documentation

8.45.4.1 ParticleDataTable & HepMC::ParticleDataTable::operator= (const ParticleDataTable &) [inline]

shallow: does not copy the entries, only makes new pointers

Definition at line 120 of file ParticleDataTable.h.

References m_data_table, and m_description.

8.45.4.2 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 128 of file ParticleDataTable.h.

References begin(), end(), insert(), merge_table(), and p.

Referenced by main().

8.45.4.3 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 243 of file ParticleDataTable.h.

References begin(), end(), insert(), and p.

Referenced by make_antiparticles_from_particles().

8.45.4.4 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 145 of file ParticleDataTable.h.

References size().

Referenced by main().

8.45.4.5 void HepMC::ParticleDataTable::delete_all () [inline]

delete all **ParticleData** (p. 190) instances in this table

deletes all **ParticleData** (p. 190) instances in this table

Examples:

example_BuildEventFromScratch.cc.

Definition at line 234 of file ParticleDataTable.h.

References clear().

Referenced by main().

8.45.4.6 void HepMC::ParticleDataTable::clear () [inline]

clears table without deleting

Definition at line 241 of file ParticleDataTable.h.

Referenced by delete_all().

8.45.4.7 ParticleData * HepMC::ParticleDataTable::operator[] (int *id*) const [inline]

return pointer to requested **ParticleData** (p. 190)

Definition at line 173 of file ParticleDataTable.h.

References find().

8.45.4.8 ParticleData * HepMC::ParticleDataTable::find (int *id*) const [inline]

return pointer to requested **ParticleData** (p. 190)

finds a **ParticleData** (p. 190) pointer corresponding to id IF it exists in the table. If not returns NULL

Definition at line 165 of file ParticleDataTable.h.

Referenced by HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table(), operator[](), and HepMC::IO_PDG_ParticleDataTable::read_entry().

8.45.4.9 int HepMC::ParticleDataTable::size () const [inline]

size of table

Definition at line 177 of file ParticleDataTable.h.

Referenced by print().

8.45.4.10 bool HepMC::ParticleDataTable::empty () const [inline]

true if the table is empty

Definition at line 181 of file ParticleDataTable.h.

8.45.4.11 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. 202)

Examples:

example_BuildEventFromScratch.cc.

Definition at line 185 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(), HepMC::IO_ExtendedAscii::read_particle_data(), and HepMC::IO_Ascii::read_particle_data().

8.45.4.12 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 193 of file ParticleDataTable.h.

References HepMC::ParticleData::pdg_id().

Referenced by HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table().

8.45.4.13 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 200 of file ParticleDataTable.h.

8.45.4.14 ParticleDataTable::iterator HepMC::ParticleDataTable::begin () [inline]

begin iteration

Definition at line 206 of file ParticleDataTable.h.

Referenced by make_antiparticles_from_particles(), merge_table(), HepMC::IO_ExtendedAscii::write_particle_data_table(), and HepMC::IO_Ascii::write_particle_data_table().

8.45.4.15 ParticleDataTable::iterator HepMC::ParticleDataTable::end () [inline]

end iteration

Definition at line 210 of file ParticleDataTable.h.

Referenced by make_antiparticles_from_particles(), merge_table(), HepMC::IO_ExtendedAscii::write_particle_data_table(), and HepMC::IO_Ascii::write_particle_data_table().

8.45.4.16 ParticleDataTable::const_iterator HepMC::ParticleDataTable::begin () const [inline]

begin const iteration

Definition at line 214 of file ParticleDataTable.h.

8.45.4.17 ParticleDataTable::const_iterator HepMC::ParticleDataTable::end () const [inline]

end const iteration

Definition at line 218 of file ParticleDataTable.h.

8.45.4.18 std::string HepMC::ParticleDataTable::description () const [inline]

table description

Definition at line 222 of file ParticleDataTable.h.

8.45.4.19 void HepMC::ParticleDataTable::set_description (std::string) [inline]

set table description

Definition at line 226 of file ParticleDataTable.h.

Referenced by HepMC::IO_PDG_ParticleDataTable::fill_particle_data_table(), HepMC::IO_-ExtendedAscii::fill_particle_data_table(), and HepMC::IO_Ascii::fill_particle_data_table().

8.45.4.20 void HepMC::ParticleDataTable::set_description (const *char*) [inline]

set table description

Definition at line 230 of file ParticleDataTable.h.

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

- **ParticleDataTable.h**

8.46 HepMC::PdfInfo Class Reference

The **PdfInfo** (p. 204) 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)
  all values 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. 204) 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
- 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).
- double **pdf2** () const
  PDF (id2, x2, Q).

- 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_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 PDF (id1, x1, Q)
- void **set_pdf2** (const double &f)
  set PDF (id2, x2, Q)

8.46.1 Detailed Description

The **PdfInfo** (p. 204) class stores PDF information.

HepMC::PdfInfo (p. 204) stores additional PDF information for a **GenEvent** (p. 46). Creation and use of this information is optional.

Definition at line 30 of file PdfInfo.h.

8.46.2 Constructor & Destructor Documentation

8.46.2.1 HepMC::PdfInfo::PdfInfo () [inline]

default constructor

Definition at line 36 of file PdfInfo.h.

8.46.2.2 HepMC::PdfInfo::PdfInfo (int *i1*, int *i2*, double *x1*, double *x2*, double *q*, double *p1*, double *p2*) [inline]

all values must be provided

Definition at line 106 of file PdfInfo.h.

8.46.2.3 HepMC::PdfInfo::~PdfInfo () [inline]

Definition at line 49 of file PdfInfo.h.

8.46.2.4 HepMC::PdfInfo::PdfInfo (PdfInfo const & *orig*) [inline]

copy constructor

Definition at line 116 of file PdfInfo.h.

8.46.3 Member Function Documentation

8.46.3.1 PdfInfo & HepMC::PdfInfo::operator= (PdfInfo const & *rhs*) [inline]

make a copy

Definition at line 126 of file PdfInfo.h.

References swap().

8.46.3.2 void HepMC::PdfInfo::swap (PdfInfo & *other*) [inline]

swap two **PdfInfo** (p. 204) objects

Definition at line 133 of file PdfInfo.h.

References m_id1, m_id2, m_pdf1, m_pdf2, m_scalePDF, m_x1, and m_x2.

Referenced by operator=().

8.46.3.3 bool HepMC::PdfInfo::operator== (const PdfInfo &) const [inline]

check for equality

equality requires that each member match

Definition at line 144 of file PdfInfo.h.

References id1(), id2(), pdf1(), pdf2(), scalePDF(), x1(), and x2().

8.46.3.4 bool HepMC::PdfInfo::operator!= (const PdfInfo &) const [inline]

check for inequality

any nonmatching member generates inequality

Definition at line 156 of file PdfInfo.h.

8.46.3.5 int HepMC::PdfInfo::id1 () const [inline]

flavour code of first parton

Definition at line 64 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.6 int HepMC::PdfInfo::id2 () const [inline]

flavour code of second parton

Definition at line 66 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.7 double HepMC::PdfInfo::x1 () const [inline]

fraction of beam momentum carried by first parton ("beam side")

Definition at line 68 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.8 double HepMC::PdfInfo::x2 () const [inline]

fraction of beam momentum carried by second parton ("target side")

Definition at line 70 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.9 double HepMC::PdfInfo::scalePDF () const [inline]

Q-scale used in evaluation of PDF's (in GeV).

Definition at line 72 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.10 double HepMC::PdfInfo::pdf1 () const [inline]

PDF (id1, x1, Q).

Definition at line 74 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.11 double HepMC::PdfInfo::pdf2 () const [inline]

PDF (id2, x2, Q).

Definition at line 76 of file PdfInfo.h.

Referenced by operator==(), and HepMC::IO_ExtendedAscii::write_pdf_info().

8.46.3.12 void HepMC::PdfInfo::set_id1 (const int & *i*) [inline]

set flavour code of first parton

Definition at line 80 of file PdfInfo.h.

8.46.3.13 void HepMC::PdfInfo::set_id2 (const int & *i*) [inline]

set flavour code of second parton

Definition at line 82 of file PdfInfo.h.

8.46.3.14 void HepMC::PdfInfo::set_x1 (const double & *f*) [inline]

set fraction of beam momentum carried by first parton ("beam side")
Definition at line 84 of file PdfInfo.h.

8.46.3.15 void HepMC::PdfInfo::set_x2 (const double & *f*) [inline]

set fraction of beam momentum carried by second parton ("target side")
Definition at line 86 of file PdfInfo.h.

8.46.3.16 void HepMC::PdfInfo::set_scalePDF (const double & *f*) [inline]

set Q-scale used in evaluation of PDF's (in GeV)
Definition at line 88 of file PdfInfo.h.

8.46.3.17 void HepMC::PdfInfo::set_pdf1 (const double & *f*) [inline]

set PDF (id1, x1, Q)
Definition at line 90 of file PdfInfo.h.

8.46.3.18 void HepMC::PdfInfo::set_pdf2 (const double & *f*) [inline]

set PDF (id2, x2, Q)
Definition at line 92 of file PdfInfo.h.
The documentation for this class was generated from the following file:

- **PdfInfo.h**

8.47 HepMC::Polarization Class Reference

The **Polarization** (p. 209) class stores theta and phi for a **GenParticle** (p. 77).

`#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. 216)*
- 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

8.47.1 Detailed Description

The **Polarization** (p. 209) class stores theta and phi for a **GenParticle** (p. 77).

HepMC::Polarization (p. 209) 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.

8.47.2 Constructor & Destructor Documentation

8.47.2.1 HepMC::Polarization::Polarization (double *theta* = 0, double *phi* = 0)

default constructor

Definition at line 11 of file Polarization.cc.

8.47.2.2 HepMC::Polarization::Polarization (const Polarization & *inpolar*)

construct from another polarization object

Definition at line 16 of file Polarization.cc.

8.47.2.3 HepMC::Polarization::Polarization (const ThreeVector & *vec3in*)

construct using the polar and azimuthal angles from a **ThreeVector** (p. 216)

Definition at line 21 of file Polarization.cc.

8.47.2.4 virtual HepMC::Polarization::~Polarization () [inline, virtual]

Definition at line 41 of file Polarization.h.

8.47.3 Member Function Documentation

8.47.3.1 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().

8.47.3.2 Polarization & HepMC::Polarization::operator= (const Polarization & *inpolar*)

make a copy

best practices implementation

Definition at line 32 of file Polarization.cc.

References swap().

8.47.3.3 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().

8.47.3.4 bool HepMC::Polarization::operator!= (const Polarization &) const [inline]

inequality results if either theta or phi differ

Definition at line 98 of file Polarization.h.

8.47.3.5 void HepMC::Polarization::print (std::ostream & *ostr* = `std::cout`) const

print theta and phi

Definition at line 39 of file Polarization.cc.

8.47.3.6 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==().

8.47.3.7 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==().

8.47.3.8 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().

8.47.3.9 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().

8.47.3.10 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().

8.47.3.11 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().

8.47.3.12 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().

8.47.4 Friends And Related Function Documentation

8.47.4.1 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**

8.48 HepMC::TempParticleMap Class Reference

TempParticleMap (p. 213) 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 &)

8.48.1 Detailed Description

TempParticleMap (p. 213) 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.

8.48.2 Member Typedef Documentation

8.48.2.1 typedef std::map<HepMC::GenParticle*,int> HepMC::TempParticleMap::TempMap

Definition at line 26 of file TempParticleMap.h.

8.48.2.2 typedef std::map<int,HepMC::GenParticle*> HepMC::TempParticleMap::TempOrderMap

Definition at line 27 of file TempParticleMap.h.

8.48.2.3 typedef TempMap::iterator HepMC::TempParticleMap::TempMapIterator

Definition at line 28 of file TempParticleMap.h.

8.48.2.4 typedef TempOrderMap::iterator HepMC::TempParticleMap::orderIterator

Definition at line 29 of file TempParticleMap.h.

8.48.3 Constructor & Destructor Documentation

8.48.3.1 HepMC::TempParticleMap::TempParticleMap () [inline]

Definition at line 31 of file TempParticleMap.h.

8.48.3.2 HepMC::TempParticleMap::~TempParticleMap () [inline]

Definition at line 34 of file TempParticleMap.h.

8.48.4 Member Function Documentation

8.48.4.1 TempMapIterator HepMC::TempParticleMap::begin () [inline]

Definition at line 36 of file TempParticleMap.h.

8.48.4.2 TempMapIterator HepMC::TempParticleMap::end () [inline]

Definition at line 37 of file TempParticleMap.h.

Referenced by end_vertex().

8.48.4.3 orderIterator HepMC::TempParticleMap::order_begin () [inline]

Definition at line 38 of file TempParticleMap.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), and HepMC::IO_Ascii::fill_next_event().

8.48.4.4 orderIterator HepMC::TempParticleMap::order_end () [inline]

Definition at line 39 of file TempParticleMap.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), and HepMC::IO_Ascii::fill_next_event().

8.48.4.5 int HepMC::TempParticleMap::end_vertex (GenParticle *) [inline]

Definition at line 51 of file TempParticleMap.h.

References end(), and p.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), and HepMC::IO_Ascii::fill_next_event().

8.48.4.6 void HepMC::TempParticleMap::addEndParticle (GenParticle *, int &) [inline]

Definition at line 59 of file TempParticleMap.h.

References p.

Referenced by HepMC::IO_ExtendedAscii::read_particle(), and HepMC::IO_Ascii::read_particle().

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

- **TempParticleMap.h**

8.49 HepMC::ThreeVector Class Reference

ThreeVector (p. 216) 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.

- double **mag** () const

  The magnitude (r in spherical coordinate system).

- void **setPhi** (double)

  Set phi keeping mag and theta constant (BaBar).

- void **setTheta** (double)

  Set theta keeping mag and phi constant (BaBar).

- double **perp2** () const

  The transverse component squared (rho^2 in cylindrical coordinate system).

- double **perp** () const

  The transverse component (rho in cylindrical coordinate system).

- **ThreeVector** & **operator**= (const **ThreeVector** &)

  make a copy

- bool **operator**== (const **ThreeVector** &) const

  equality

- bool **operator!**= (const **ThreeVector** &) const

  inequality

8.49.1 Detailed Description

ThreeVector (p. 216) is a simple representation of a position or displacement 3 vector.

For compatibility with existing code, the basic expected geometrical access methods are povided. Also, there is a templated constructor that will take another vector (HepLorentzVector, GenVector, ...) which must have the following methods: **x()** (p. 218), **y()** (p. 218), **z()** (p. 218).

Definition at line 132 of file SimpleVector.h.

8.49.2 Constructor & Destructor Documentation

8.49.2.1 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 137 of file SimpleVector.h.

8.49.2.2 HepMC::ThreeVector::ThreeVector () [inline]

Definition at line 140 of file SimpleVector.h.

8.49.2.3 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 146 of file SimpleVector.h.

8.49.2.4 HepMC::ThreeVector::ThreeVector (const ThreeVector & *v*) [inline]

copy constructor

Definition at line 151 of file SimpleVector.h.

8.49.3 Member Function Documentation

8.49.3.1 void HepMC::ThreeVector::swap (ThreeVector & *other*)

swap

8.49.3.2 double HepMC::ThreeVector::x () const [inline]

return x

Definition at line 156 of file SimpleVector.h.

Referenced by main().

8.49.3.3 double HepMC::ThreeVector::y () const [inline]

return y

Definition at line 157 of file SimpleVector.h.

Referenced by main().

8.49.3.4 double HepMC::ThreeVector::z () const [inline]

return z

Definition at line 158 of file SimpleVector.h.

Referenced by main().

8.49.3.5 void HepMC::ThreeVector::setX (double *x*) [inline]

set x

Definition at line 160 of file SimpleVector.h.

Referenced by main().

8.49.3.6 void HepMC::ThreeVector::setY (double *y*) [inline]

set y

Definition at line 161 of file SimpleVector.h.

Referenced by main().

8.49.3.7 void HepMC::ThreeVector::setZ (double *z*) [inline]

set z

Definition at line 162 of file SimpleVector.h.

Referenced by main().

8.49.3.8 void HepMC::ThreeVector::set (double *x*, double *y*, double *z*) [inline]

set x, y, and z

Referenced by main().

8.49.3.9 double HepMC::ThreeVector::phi () const [inline]

The azimuth angle.

Referenced by main(), and HepMC::Polarization::set_normal3d().

8.49.3.10 double HepMC::ThreeVector::theta () const [inline]

The polar angle.

Referenced by main(), and HepMC::Polarization::set_normal3d().

8.49.3.11 double HepMC::ThreeVector::r () const [inline]

The magnitude.

Referenced by main().

8.49.3.12 double HepMC::ThreeVector::mag () const [inline]

The magnitude (r in spherical coordinate system).

Referenced by main().

8.49.3.13 void HepMC::ThreeVector::setPhi (double) [inline]

Set phi keeping mag and theta constant (BaBar).

Referenced by main(), and HepMC::Polarization::normal3d().

8.49.3.14 void HepMC::ThreeVector::setTheta (double) [inline]

Set theta keeping mag and phi constant (BaBar).

Referenced by main(), and HepMC::Polarization::normal3d().

8.49.3.15 double HepMC::ThreeVector::perp2 () const [inline]

The transverse component squared (rho^2 in cylindrical coordinate system).

Referenced by main().

8.49.3.16 double HepMC::ThreeVector::perp () const [inline]

The transverse component (rho in cylindrical coordinate system).

Referenced by main().

8.49.3.17 ThreeVector& HepMC::ThreeVector::operator= (const ThreeVector &) [inline]

make a copy

8.49.3.18 bool HepMC::ThreeVector::operator== (const ThreeVector &) const [inline]

equality

8.49.3.19 bool HepMC::ThreeVector::operator!= (const ThreeVector &) const [inline]

inequality

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

- **SimpleVector.h**

8.50 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

8.50.1 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.

8.50.2 Member Typedef Documentation

8.50.2.1 typedef std::vector<double>::iterator HepMC::WeightContainer::iterator

iterator for the weight container

Definition at line 71 of file WeightContainer.h.

8.50.2.2 typedef std::vector<double>::const_iterator HepMC::WeightContainer::const_iterator

const iterator for the weight container

Definition at line 73 of file WeightContainer.h.

8.50.3 Constructor & Destructor Documentation

8.50.3.1 HepMC::WeightContainer::WeightContainer (unsigned int *n* = 0, const double & *value* = 0.) [inline]

default constructor

Definition at line 91 of file WeightContainer.h.

8.50.3.2 HepMC::WeightContainer::WeightContainer (const std::vector< double > & *weights*) [inline]

construct from a vector of weights

Definition at line 96 of file WeightContainer.h.

8.50.3.3 HepMC::WeightContainer::WeightContainer (const WeightContainer & *in*) [inline]

copy

Definition at line 100 of file WeightContainer.h.

8.50.3.4 HepMC::WeightContainer::~WeightContainer () [inline, virtual]

Definition at line 104 of file WeightContainer.h.

8.50.4 Member Function Documentation

8.50.4.1 void HepMC::WeightContainer::swap (WeightContainer & *other*) [inline]

swap

Definition at line 106 of file WeightContainer.h.

References m_weights.

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

8.50.4.2 WeightContainer & HepMC::WeightContainer::operator= (const WeightContainer &) [inline]

copy

best practices implementation

Definition at line 110 of file WeightContainer.h.

References swap().

8.50.4.3 WeightContainer & HepMC::WeightContainer::operator= (const std::vector< double > & *in*) [inline]

copy

best practices implementation

Definition at line 119 of file WeightContainer.h.

References swap().

8.50.4.4 void HepMC::WeightContainer::print (std::ostream & *ostr* = std::cout) const [inline]

print weights

Definition at line 126 of file WeightContainer.h.

References begin(), and end().

8.50.4.5 int HepMC::WeightContainer::size () const [inline]

size of weight container

Definition at line 135 of file WeightContainer.h.

Referenced by HepMC::GenVertex::print(), HepMC::GenEvent::print(), HepMC::IO_Extended-Ascii::write_event(), HepMC::IO_AsciiParticles::write_event(), and HepMC::IO_Ascii::write_-event().

8.50.4.6 bool HepMC::WeightContainer::empty () const [inline]

return true if weight container is empty

Definition at line 137 of file WeightContainer.h.

8.50.4.7 void HepMC::WeightContainer::push_back (const double &) [inline]

push onto weight container

Definition at line 139 of file WeightContainer.h.

8.50.4.8 void HepMC::WeightContainer::pop_back () [inline]

pop from weight container

Definition at line 142 of file WeightContainer.h.

8.50.4.9 void HepMC::WeightContainer::clear () [inline]

clear the weight container

Definition at line 144 of file WeightContainer.h.

8.50.4.10 double & HepMC::WeightContainer::operator[] (unsigned int n) [inline]

access the weight container

Definition at line 146 of file WeightContainer.h.

8.50.4.11 const double & HepMC::WeightContainer::operator[] (unsigned int n) const [inline]

access the weight container

Definition at line 149 of file WeightContainer.h.

8.50.4.12 double & HepMC::WeightContainer::front () [inline]

returns the first element

Definition at line 152 of file WeightContainer.h.

8.50.4.13 const double & HepMC::WeightContainer::front () const [inline]

returns the first element

Definition at line 154 of file WeightContainer.h.

8.50.4.14 double & HepMC::WeightContainer::back () [inline]

returns the last element

Definition at line 157 of file WeightContainer.h.

8.50.4.15 const double & HepMC::WeightContainer::back () const [inline]

returns the last element

Definition at line 159 of file WeightContainer.h.

8.50.4.16 WeightContainer::iterator HepMC::WeightContainer::begin () [inline]

begining of the weight container

Definition at line 162 of file WeightContainer.h.

Referenced by print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_Ascii-Particles::write_event(), and HepMC::IO_Ascii::write_event().

8.50.4.17 WeightContainer::iterator HepMC::WeightContainer::end () [inline]

end of the weight container

Definition at line 165 of file WeightContainer.h.

Referenced by print(), HepMC::GenVertex::print(), HepMC::GenEvent::print(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_AsciiParticles::write_event(), and HepMC::IO_Ascii::write_event().

8.50.4.18 WeightContainer::const_iterator HepMC::WeightContainer::begin () const [inline]

begining of the weight container

Definition at line 168 of file WeightContainer.h.

8.50.4.19 WeightContainer::const_iterator HepMC::WeightContainer::end () const [inline]

end of the weight container

Definition at line 171 of file WeightContainer.h.

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

- **WeightContainer.h**

Chapter 9

HepMC File Documentation

9.1 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

9.2 example_BuildEventFromScratch.cc File Reference

```
#include <iostream>
#include "VectorConversion.h"
#include "HepMC/GenEvent.h"
#include "HepMC/ParticleDataTable.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Namespaces

- namespace **CLHEP**

Functions

- int **main** ()

9.2.1 Function Documentation

9.2.1.1 int main ()

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, **example_MyHerwig.cc**, **example_MyPythia.cc**, **example_MyPythiaOnlyToHepMC.cc**, **example_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, **example_PythiaParticle.cc**, and **example_UsingIterators.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::clifetime_from_width(), HepMC::ParticleDataTable::delete_all(), HepMC::ParticleDataTable::insert(), p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::print(), HepMC::ParticleDataTable::print(), HepMC::GenEvent::set_signal_process_vertex(), and SVtoLV().

9.3 example_EventSelection.cc File Reference

```
#include "HepMC/IO_Ascii.h"
#include "HepMC/GenEvent.h"
```

Classes

- class **IsGoodEvent**
  example class

Functions

- int **main** ()

9.3.1 Function Documentation

9.3.1.1 int main ()

Definition at line 37 of file example_EventSelection.cc.

References HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

9.4 example_MyHerwig.cc File Reference

```
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Functions

- int **main** ()

9.4.1 Function Documentation

9.4.1.1 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 23 of file example_MyHerwig.cc.

References hwbgen, hwbmch, hwcdec, hwcfor, hwdhad, hwdhob, hwdhvy, hwefin, hweini, hwepro, hwevnt, hwigin, hwmevt, hwproc, hwufne, hwuinc, hwuine, HepMC::GenEvent::print(), HepMC::HEPEVT_Wrapper::print_hepevt(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), and HepMC::HEPEVT_Wrapper::set_sizeof_real().

9.5 example_MyPythia.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_Ascii.h"
#include "HepMC/IO_ExtendedAscii.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

9.5.1 Function Documentation

9.5.1.1 int main ()

To Compile: go to the **HepMC** (p. 19) directory and type: gmake examples/example_MyPythia.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 Pythia 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 29 of file example_MyPythia.cc.

References initPythia(), pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::GenEvent::set_signal_process_id(), and HepMC::HEPEVT_Wrapper::set_sizeof_real().

9.6 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** ()

9.6.1 Function Documentation

9.6.1.1 int main ()

Definition at line 23 of file example_MyPythiaOnlyToHepMC.cc.

References initPythia(), pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), and HepMC::HEPEVT_Wrapper::set_sizeof_real().

9.7 example_MyPythiaRead.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_Ascii.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

9.7.1 Function Documentation

9.7.1.1 int main ()

Definition at line 28 of file example_MyPythiaRead.cc.

References HepMC::GenEvent::event_number(), initPythia(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), and HepMC::HEPEVT_Wrapper::set_sizeof_real().

9.8 example_MyPythiaWithEventSelection.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Classes

- class **IsGoodEventMyPythia**
  example class

Functions

- int **main** ()

9.8.1 Function Documentation

9.8.1.1 int main ()

Definition at line 45 of file example_MyPythiaWithEventSelection.cc.

References initPythia(), pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), and HepMC::HEPEVT_Wrapper::set_sizeof_real().

9.9 example_PythiaParticle.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

9.9.1 Function Documentation

9.9.1.1 int main ()

Definition at line 29 of file example_PythiaParticle.cc.

References initPythia(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), and HepMC::HEPEVT_Wrapper::set_sizeof_real().

9.10 example_ReadPDGtable.cc File Reference

```
#include "HepMC/IO_PDG_ParticleDataTable.h"
```

Functions

- int **main** ()

9.10.1 Function Documentation

9.10.1.1 int main ()

Definition at line 16 of file example_ReadPDGtable.cc.

References HepMC::IO_PDG_ParticleDataTable::add_quarks_to_table(), HepMC::ParticleDataTable::delete_all(), HepMC::ParticleDataTable::make_antiparticles_from_particles(), HepMC::ParticleDataTable::print(), HepMC::IO_PDG_ParticleDataTable::rdstate(), and HepMC::IO_BaseClass::read_particle_data_table().

9.11 example_UsingIterators.cc File Reference

```
#include "HepMC/IO_Ascii.h"
#include "HepMC/GenEvent.h"
#include <math.h>
#include <algorithm>
#include <list>
```

Classes

- class **IsPhoton**
  example class
- class **IsW_Boson**
  example class
- class **IsFinalState**
  example class

Functions

- int **main** ()

9.11.1 Function Documentation

9.11.1.1 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_Ascii::rdstate(), HepMC::IO_BaseClass::read_next_event(), v, HepMC::GenEvent::vertices_begin(), and HepMC::GenEvent::vertices_end().

9.12 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

9.13 Flow.h File Reference

```
#include <iostream>
#include <map>
#include <vector>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::Flow**
  The flow object.

9.14 GenEvent.cc File Reference

```
#include "HepMC/GenEvent.h"
#include "HepMC/Version.h"
```

Namespaces

- namespace **HepMC**

9.15 GenEvent.h File Reference

```
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include "HepMC/WeightContainer.h"
#include "HepMC/HeavyIon.h"
#include "HepMC/PdfInfo.h"
#include <map>
#include <vector>
#include <algorithm>
#include <iostream>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::GenEvent**
  *The **GenEvent** (p. 46) 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

9.16 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

9.17 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. 77) class contains information about generated particles.*

Defines

- #define **hepmc_uint64_t** uint64_t

9.17.1 Define Documentation

9.17.1.1 #define hepmc_uint64_t uint64_t

Definition at line 37 of file GenParticle.h.

9.18 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

9.19 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. 87) 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

9.20 HeavyIon.h File Reference

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::HeavyIon**

  *The **HeavyIon** (p. 112) class stores information about heavy ions.*

9.21 HEPEVT_Wrapper.cc File Reference

```
#include "HepMC/HEPEVT_Wrapper.h"
```

Namespaces

- namespace **HepMC**

9.22 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_**

9.22.1 Define Documentation

9.22.1.1 #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().

9.22.1.2 #define HEPEVT_EntriesAllocation 10000

Definition at line 4 of file HEPEVT_Wrapper.h.

9.22.2 Variable Documentation

9.22.2.1 char data[hepevt_bytes_allocation]

Definition at line 81 of file HEPEVT_Wrapper.h.

9.22.2.2 struct { ... } hepevt_

9.22.2.3 const unsigned int hepevt_bytes_allocation

Initial value:

```
sizeof(long int) * ( 2 + 4 * 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().

9.23 HepMC_CLHEP20.h File Reference

Typedefs

- typedef **CLHEP::HepLorentzVector HepLorentzVector**
- typedef HepGeom::Normal3D< double > **HepNormal3D**
- typedef HepGeom::Point3D< double > **HepPoint3D**

9.23.1 Typedef Documentation

9.23.1.1 typedef CLHEP::HepLorentzVector HepLorentzVector

Examples:

example_BuildEventFromScratch.cc.

Definition at line 13 of file HepMC_CLHEP20.h.

9.23.1.2 typedef HepGeom::Normal3D<double> HepNormal3D

Definition at line 16 of file HepMC_CLHEP20.h.

9.23.1.3 typedef HepGeom::Point3D<double> HepPoint3D

Definition at line 17 of file HepMC_CLHEP20.h.

9.24 HerwigWrapper.h File Reference

```
#include "HepMC/HerwigWrapper6_4.h"
```

9.25 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 **GENSOF**
  } **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 **ENSOF**
  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 **PRSOF**
  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_**

9.25.1 Define Documentation

9.25.1.1 #define hwbgen hwbgen_

Examples:

example_MyHerwig.cc.

Definition at line 87 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.2 #define hwbmch hwbmch_

Examples:

example_MyHerwig.cc.

Definition at line 32 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.3 #define hwcdec hwcdec_

Examples:

example_MyHerwig.cc.

Definition at line 90 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.4 #define hwcfor hwcfor_

Examples:

example_MyHerwig.cc.

Definition at line 89 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.5 #define hwdhad hwdhad_

Examples:

example_MyHerwig.cc.

Definition at line 91 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.6 #define hwdhob hwdhob_

Examples:

example_MyHerwig.cc.

Definition at line 88 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.7 #define hwdhvy hwdhvy_

Examples:

example_MyHerwig.cc.

Definition at line 92 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.8 #define hwefin hwefin_

Examples:

example_MyHerwig.cc.

Definition at line 95 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.9 #define hwegup hwegup_

Definition at line 100 of file HerwigWrapper6_4.h.

9.25.1.10 #define hweini hweini_

Examples:

example_MyHerwig.cc.

Definition at line 83 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.11 #define hwepro hwepro_

Examples:

example_MyHerwig.cc.

Definition at line 85 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.12 #define hwevnt hwevnt_

Examples:

example_MyHerwig.cc.

Definition at line 46 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.13 #define hwigin hwigin_

Examples:

example_MyHerwig.cc.

Definition at line 79 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.14 #define hwigup hwigup_

Definition at line 80 of file HerwigWrapper6_4.h.

9.25.1.15 #define hwmevt hwmevt_

Examples:

example_MyHerwig.cc.

Definition at line 93 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.16 #define hwpram hwpram_

Definition at line 74 of file HerwigWrapper6_4.h.

9.25.1.17 #define hwproc hwproc_

Examples:

example_MyHerwig.cc.

Definition at line 23 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.18 #define hwudat hwudat_

9.25.1.19 #define hwudpr hwudpr_

Definition at line 97 of file HerwigWrapper6_4.h.

9.25.1.20 #define hwuepr hwuepr_

Definition at line 98 of file HerwigWrapper6_4.h.

9.25.1.21 #define hwufne hwufne_

Examples:

example_MyHerwig.cc.

Definition at line 94 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.22 #define hwuinc hwuinc_

Examples:

example_MyHerwig.cc.

Definition at line 81 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.23 #define hwuine hwuine_

Examples:

example_MyHerwig.cc.

Definition at line 84 of file HerwigWrapper6_4.h.
Referenced by main().

9.25.1.24 #define hwupro hwupro_

Definition at line 86 of file HerwigWrapper6_4.h.

9.25.1.25 #define hwupup hwupup_

Definition at line 99 of file HerwigWrapper6_4.h.

9.25.1.26 #define hwusta hwusta_

Definition at line 82 of file HerwigWrapper6_4.h.

9.25.2 Variable Documentation

9.25.2.1 double AFCH[2][16]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.2 double ALPHEM

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.3 double AVWGT

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.4 int AZSOFT

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.5 int AZSPIN

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.6 double B1LIM

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.7 double BETAF

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.8 double BTCLM

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.9 double CAFAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.10 double CFFAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.11 int CLDIR[2]

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.12 double CLMAX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.13 double CLPOW

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.14 double CLSMR[2]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.15 double CSPEED

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.16 double EBEAM1

Definition at line 19 of file HerwigWrapper6_4.h.

9.25.2.17 double EBEAM2

Definition at line 19 of file HerwigWrapper6_4.h.

9.25.2.18 double EFFMIN

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.19 double ENSOF

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.20 double ET2MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.21 double ETAMIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.22 double EVWGT

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.23 double F0MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.24 double F1MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.25 double F2MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.26 double GAMH

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.27 double GAMW

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.28 double GAMWT

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.29 double GAMZ

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.30 double GAMZP

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.31 double GCUTME

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.32 int GENSOF

Definition at line 43 of file HerwigWrapper6_4.h.

9.25.2.33 double GEV2NB

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.34 double H1MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.35 int HARDME

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.36 const int herwig_hepevt_size = 4000

Definition at line 37 of file HerwigWrapper6_4.h.

9.25.2.37 struct { ... } hwbmch_

9.25.2.38 struct { ... } hwevnt_

9.25.2.39 struct { ... } hwpram_

9.25.2.40 struct { ... } hwproc_

9.25.2.41 int IDHW[herwig_hepevt_size]

Definition at line 41 of file HerwigWrapper6_4.h.

9.25.2.42 int IERROR

Definition at line 41 of file HerwigWrapper6_4.h.

9.25.2.43 int IOP4JT[2]

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.44 int IOPREM

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.45 int IPRINT

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.46 int IPROC

Definition at line 20 of file HerwigWrapper6_4.h.

9.25.2.47 int ISPAC

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.48 int ISTAT

Definition at line 41 of file HerwigWrapper6_4.h.

9.25.2.49 int LRSUD

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.50 int LWEVT

Definition at line 41 of file HerwigWrapper6_4.h.

9.25.2.51 int LWSUD

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.52 int MAXER

Definition at line 41 of file HerwigWrapper6_4.h.

9.25.2.53 int MAXEV

Definition at line 20 of file HerwigWrapper6_4.h.

9.25.2.54 int MAXPR

Definition at line 41 of file HerwigWrapper6_4.h.

9.25.2.55 int MODPDF[2]

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.56 int NBTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.57 int NCOLO

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.58 int NCTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.59 int NDTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.60 int NETRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.61 int NFLAV

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.62 int NGSPL

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.63 int NOSPAC

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.64 int NOWGT

Definition at line 42 of file HerwigWrapper6_4.h.

9.25.2.65 int NPRFMT

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.66 int NRN[2]

Definition at line 42 of file HerwigWrapper6_4.h.

9.25.2.67 int NSTRU

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.68 int NSTRY

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.69 int NUMER

Definition at line 42 of file HerwigWrapper6_4.h.

9.25.2.70 int NUMERU

Definition at line 42 of file HerwigWrapper6_4.h.

9.25.2.71 int NWGTS

Definition at line 42 of file HerwigWrapper6_4.h.

9.25.2.72 int NZBIN

Definition at line 68 of file HerwigWrapper6_4.h.

9.25.2.73 double OMHMIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.74 char PART1[8]

Definition at line 29 of file HerwigWrapper6_4.h.

9.25.2.75 char PART2[8]

Definition at line 29 of file HerwigWrapper6_4.h.

9.25.2.76 double PBEAM1

Definition at line 19 of file HerwigWrapper6_4.h.

9.25.2.77 double PBEAM2

Definition at line 19 of file HerwigWrapper6_4.h.

9.25.2.78 double PDIQK

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.79 double PGSMX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.80 double PGSPL[4]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.81 double PH3MIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.82 double PHIMIX

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.83 double PIFAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.84 int PRNDEC

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.85 int PRNDEF

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.86 int PRNTEX

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.87 int PRNWEB

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.88 double PRSOF

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.89 int PRVTX

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.90 double PSPLT[2]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.91 double PTRMS

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.92 double PXRMS

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.93 double QCDL3

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.94 double QCDL5

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.95 double QCDLAM

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.96 double QDIQK

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.97 double QFCH[16]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.98 double QG

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.99 double QSPAC

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.100 double QV

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.101 double SCABI

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.102 int SOFTME

Definition at line 70 of file HerwigWrapper6_4.h.

9.25.2.103 double SWEIN

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.104 double TLOUT

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.105 double TMTOP

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.106 double VCKM[3][3]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.107 double VFCH[2][16]

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.108 double VGCUT

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.109 double VPCUT

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.110 double VQCUT

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.111 double WBIGST

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.112 double WGTMAX

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.113 double WGTSUM

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.114 double WSQSUM

Definition at line 40 of file HerwigWrapper6_4.h.

9.25.2.115 double ZBINM

Definition at line 62 of file HerwigWrapper6_4.h.

9.25.2.116 int ZPRIME

Definition at line 70 of file HerwigWrapper6_4.h.

9.26 initPythia.cc File Reference

```
#include "HepMC/PythiaWrapper.h"
```

Functions

- void **initPythia** ()

9.26.1 Function Documentation

9.26.1.1 void initPythia ()

Examples:

example_MyPythia.cc, **example_MyPythiaOnlyToHepMC.cc**, **example_MyPythiaRead.cc**, **example_MyPythiaWithEventSelection.cc**, and **example_PythiaParticle.cc**.

Definition at line 11 of file initPythia.cc.

References pydat2, pydatr, pypars, and pysubs.

Referenced by main().

9.27 IO_Ascii.cc File Reference

```
#include "HepMC/IO_Ascii.h"
#include "HepMC/GenEvent.h"
#include "HepMC/ParticleDataTable.h"
#include "HepMC/Version.h"
```

Namespaces

- namespace **HepMC**

9.28 IO_Ascii.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/TempParticleMap.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_Ascii**
  IO_Ascii *(*p. *133) is used to read or write from an ascii file.*

9.29 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**

9.30 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

9.31 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. 144*)*

9.32 IO_ExtendedAscii.cc File Reference

```
#include "HepMC/IO_ExtendedAscii.h"
#include "HepMC/GenEvent.h"
#include "HepMC/ParticleDataTable.h"
#include "HepMC/HeavyIon.h"
#include "HepMC/PdfInfo.h"
#include "HepMC/Version.h"
```

Namespaces

- namespace **HepMC**

9.33 IO_ExtendedAscii.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/TempParticleMap.h"
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::IO_ExtendedAscii**
  IO_ExtendedAscii *(p. 148) also deals with* ***HeavyIon*** *(p. 112) and* ***PdfInfo*** *(p. 204).*

9.34 IO_HEPEVT.cc File Reference

```
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include <cstdio>
```

Namespaces

- namespace **HepMC**

9.35 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.

9.36 IO_HERWIG.cc File Reference

```
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include <cstdio>
```

Namespaces

- namespace **HepMC**

9.37 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. 162) is used to get Herwig information.*

9.38 IO_PDG_ParticleDataTable.cc File Reference

```
#include <ctype.h>
#include <string>
#include <vector>
#include <cstdlib>
#include "HepMC/IO_PDG_ParticleDataTable.h"
```

Namespaces

- namespace **HepMC**

9.39 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. *197) IO method*

9.40 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

9.41 IsGoodEvent.h File Reference

Classes

- class **IsGoodEvent**
  example class

9.42 list_of_examples.cc File Reference

9.43 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

9.44 ParticleData.h File Reference

```
#include <iostream>
#include <string>
#include <cmath>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::ParticleData**
  *an example **ParticleData** (p. 190) class*

Functions

- double **HepMC::clifetime_from_width** (double width)
  set lifetime from width

Variables

- static const double **HepMC::HepMC_hbarc**
  *hbar * c -> calculated with units of [mm*GeV]*

9.45 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. *197**) class*

9.46 PdfInfo.h File Reference

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::PdfInfo**

  *The **PdfInfo** (p. 204) class stores PDF information.*

9.47 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

9.48 Polarization.h File Reference

```
#include "HepMC/SimpleVector.h"
#include <iostream>
#include <cmath>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::Polarization**
  *The **Polarization** (p. 209) class stores theta and phi for a **GenParticle** (p. 77).*

Variables

- static const double **HepMC::HepMC_pi** = 3.14159265358979323846

9.49 PythiaHelper.h File Reference

Functions

- void **initPythia** ()

9.49.1 Function Documentation

9.49.1.1 void initPythia ()

Definition at line 11 of file initPythia.cc.

References pydat2, pydatr, pypars, and pysubs.

Referenced by main().

9.50 PythiaWrapper.h File Reference

```
#include "HepMC/PythiaWrapper6_2.h"
```

9.51 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 **mdcy** [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_**

9.51.1 Define Documentation

9.51.1.1 #define initpydata initpydata_

Definition at line 26 of file PythiaWrapper5_720.h.

9.51.1.2 #define ludat1 ludat1_

Definition at line 47 of file PythiaWrapper5_720.h.

9.51.1.3 #define ludat2 ludat2_

Definition at line 53 of file PythiaWrapper5_720.h.

9.51.1.4 #define ludat3 ludat3_

Definition at line 60 of file PythiaWrapper5_720.h.

9.51.1.5 #define ludata ludata_

Definition at line 109 of file PythiaWrapper5_720.h.

9.51.1.6 #define ludatr ludatr_

Definition at line 66 of file PythiaWrapper5_720.h.

9.51.1.7 #define luhepc luhepc_

Definition at line 104 of file PythiaWrapper5_720.h.

9.51.1.8 #define lujets lujets_

Definition at line 39 of file PythiaWrapper5_720.h.

9.51.1.9 #define lulist lulist_

Definition at line 106 of file PythiaWrapper5_720.h.

9.51.1.10 #define pydata pydata_

9.51.1.11 #define pyevnt pyevnt_

Definition at line 108 of file PythiaWrapper5_720.h.

9.51.1.12 #define pyinit pyinit_

Definition at line 105 of file PythiaWrapper5_720.h.

9.51.1.13 #define pyint1 pyint1_

Definition at line 86 of file PythiaWrapper5_720.h.

9.51.1.14 #define pyint2 pyint2_

Definition at line 93 of file PythiaWrapper5_720.h.

9.51.1.15 #define pyint5 pyint5_

Definition at line 99 of file PythiaWrapper5_720.h.

9.51.1.16 #define pypars pypars_

Examples:

example_MyPythia.cc, **example_MyPythiaOnlyToHepMC.cc**, and **example_MyPythiaWithEventSelection.cc**.

Definition at line 80 of file PythiaWrapper5_720.h.

Referenced by initPythia(), and main().

9.51.1.17 #define pystat pystat_

Definition at line 107 of file PythiaWrapper5_720.h.

9.51.1.18 #define pysubs pysubs_

Definition at line 72 of file PythiaWrapper5_720.h.

Referenced by initPythia().

9.51.2 Variable Documentation

9.51.2.1 float brat[2000]

Definition at line 57 of file PythiaWrapper5_720.h.

9.51.2.2 float ckin[200]

Definition at line 70 of file PythiaWrapper5_720.h.

9.51.2.3 float coef[20][200]

Definition at line 90 of file PythiaWrapper5_720.h.

9.51.2.4 int icol[2][4][40]

Definition at line 91 of file PythiaWrapper5_720.h.

9.51.2.5 int iset[200]

Definition at line 89 of file PythiaWrapper5_720.h.

9.51.2.6 int k[5][pyjets_maxn]

Definition at line 36 of file PythiaWrapper5_720.h.

9.51.2.7 int kchg[3][500]

Definition at line 50 of file PythiaWrapper5_720.h.

9.51.2.8 int kfdp[5][2000]

Definition at line 58 of file PythiaWrapper5_720.h.

9.51.2.9 int kfin[81][2]

Definition at line 69 of file PythiaWrapper5_720.h.

9.51.2.10 int kfpr[2][200]

Definition at line 89 of file PythiaWrapper5_720.h.

9.51.2.11 struct { ... } ludat1_

9.51.2.12 struct { ... } ludat2_

9.51.2.13 struct { ... } ludat3_

9.51.2.14 struct { ... } ludatr_

9.51.2.15 struct { ... } lujets_

9.51.2.16 int mdcy[3][500]

Definition at line 56 of file PythiaWrapper5_720.h.

9.51.2.17 int mdme[2][2000]

Definition at line 56 of file PythiaWrapper5_720.h.

9.51.2.18 int mint[400]

Definition at line 83 of file PythiaWrapper5_720.h.

9.51.2.19 int mrlu[6]

Definition at line 63 of file PythiaWrapper5_720.h.

9.51.2.20 int msel

Definition at line 69 of file PythiaWrapper5_720.h.

9.51.2.21 int msti[200]

Definition at line 77 of file PythiaWrapper5_720.h.

9.51.2.22 int mstj[200]

Definition at line 44 of file PythiaWrapper5_720.h.

9.51.2.23 int mstp[200]

Definition at line 75 of file PythiaWrapper5_720.h.

9.51.2.24 int mstu[200]

Definition at line 42 of file PythiaWrapper5_720.h.

9.51.2.25 int msub[200]

Definition at line 69 of file PythiaWrapper5_720.h.

9.51.2.26 int n

Definition at line 36 of file PythiaWrapper5_720.h.

9.51.2.27 int ngen[3][201][3]

Definition at line 96 of file PythiaWrapper5_720.h.

9.51.2.28 float p[5][pyjets_maxn]

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, **example_MyPythiaWithEventSelection.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_HERWIG::build_particle(), HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::GenVertex::edge_iterator::edge_iterator(), HepMC::TempParticleMap::end_vertex(), HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), HepMC::IO_HERWIG::find_in_map(), HepMC::GenEvent::GenEvent(), IsPhoton(), IsWBoson(),

main(), HepMC::ParticleDataTable::make_antiparticles_from_particles(), HepMC::ParticleDataTable::merge_table(), HepMC::not_in_vector(), IsFinalState::operator()(), IsW_Boson::operator()(), IsPhoton::operator()(), IsGoodEventMyPythia::operator()(), IsGoodEvent::operator()(), HepMC::GenVertex::edge_iterator::operator=(), HepMC::IO_ExtendedAscii::read_particle(), HepMC::IO_Ascii::read_particle(), HepMC::GenEvent::remove_barcode(), HepMC::GenEvent::set_barcode(), HepMC::GenEvent::set_pdf_info(), HepMC::GenEvent::valid_beam_particles(), HepMC::IO_HEPEVT::write_event(), HepMC::IO_ExtendedAscii::write_particle(), and HepMC::IO_Ascii::write_particle().

9.51.2.29 float parf[2000]

Definition at line 51 of file PythiaWrapper5_720.h.

9.51.2.30 float pari[200]

Definition at line 78 of file PythiaWrapper5_720.h.

9.51.2.31 float parj[200]

Definition at line 45 of file PythiaWrapper5_720.h.

9.51.2.32 float parp[200]

Definition at line 76 of file PythiaWrapper5_720.h.

9.51.2.33 float paru[200]

Definition at line 43 of file PythiaWrapper5_720.h.

9.51.2.34 float pmas[4][500]

Definition at line 51 of file PythiaWrapper5_720.h.

9.51.2.35 struct { ... } pyint1_

9.51.2.36 struct { ... } pyint2_

9.51.2.37 struct { ... } pyint5_

9.51.2.38 struct { ... } pypars_

9.51.2.39 struct { ... } pysubs_

9.51.2.40 float rrlu[100]

Definition at line 64 of file PythiaWrapper5_720.h.

9.51.2.41 float v[5][pyjets_maxn]

Examples:

example_UsingIterators.cc.

Definition at line 37 of file PythiaWrapper5_720.h.

Referenced by HepMC::IO_ExtendedAscii::fill_next_event(), HepMC::IO_Ascii::fill_next_event(), HepMC::IO_Ascii::find_in_map(), HepMC::GenEvent::GenEvent(), main(), HepMC::IO_ExtendedAscii::read_vertex(), HepMC::IO_Ascii::read_vertex(), HepMC::GenEvent::remove_barcode(), HepMC::GenEvent::set_barcode(), SVtoLV(), HepMC::IO_HEPEVT::write_event(), HepMC::IO_ExtendedAscii::write_event(), HepMC::IO_Ascii::write_event(), HepMC::IO_ExtendedAscii::write_vertex(), and HepMC::IO_Ascii::write_vertex().

9.51.2.42 float vckm[4][4]

Definition at line 51 of file PythiaWrapper5_720.h.

9.51.2.43 float vint[400]

Definition at line 84 of file PythiaWrapper5_720.h.

9.51.2.44 float xsec[3][201]

Definition at line 97 of file PythiaWrapper5_720.h.

9.52 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 **mdcy** [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_**

9.52.1 Define Documentation

9.52.1.1 #define initpydata initpydata_

Definition at line 27 of file PythiaWrapper6_152.h.

9.52.1.2 #define pydat1 pydat1_

Definition at line 48 of file PythiaWrapper6_152.h.

9.52.1.3 #define pydat2 pydat2_

Definition at line 54 of file PythiaWrapper6_152.h.
Referenced by initPythia().

9.52.1.4 #define pydat3 pydat3_

Definition at line 61 of file PythiaWrapper6_152.h.

9.52.1.5 #define pydata pydata_

9.52.1.6 #define pydatr pydatr_

Definition at line 67 of file PythiaWrapper6_152.h.
Referenced by initPythia().

9.52.1.7 #define pyevnt pyevnt_

Definition at line 109 of file PythiaWrapper6_152.h.

9.52.1.8 #define pyhepc pyhepc_

Definition at line 105 of file PythiaWrapper6_152.h.

9.52.1.9 #define pyinit pyinit_

Definition at line 106 of file PythiaWrapper6_152.h.

9.52.1.10 #define pyint1 pyint1_

Definition at line 87 of file PythiaWrapper6_152.h.

9.52.1.11 #define pyint2 pyint2_

Definition at line 94 of file PythiaWrapper6_152.h.

9.52.1.12 #define pyint5 pyint5_

Definition at line 100 of file PythiaWrapper6_152.h.

9.52.1.13 #define pyjets pyjets_

Definition at line 40 of file PythiaWrapper6_152.h.

9.52.1.14 #define pylist pylist_

Definition at line 107 of file PythiaWrapper6_152.h.

9.52.1.15 #define pypars pypars_

Definition at line 81 of file PythiaWrapper6_152.h.

9.52.1.16 #define pystat pystat_

Definition at line 108 of file PythiaWrapper6_152.h.

9.52.1.17 #define pysubs pysubs_

Definition at line 73 of file PythiaWrapper6_152.h.

9.52.2 Variable Documentation

9.52.2.1 double brat[4000]

Definition at line 58 of file PythiaWrapper6_152.h.

9.52.2.2 double ckin[200]

Definition at line 71 of file PythiaWrapper6_152.h.

9.52.2.3 double coef[20][500]

Definition at line 91 of file PythiaWrapper6_152.h.

9.52.2.4 int icol[2][4][40]

Definition at line 92 of file PythiaWrapper6_152.h.

9.52.2.5 int iset[500]

Definition at line 90 of file PythiaWrapper6_152.h.

9.52.2.6 int k[5][pyjets_maxn]

Definition at line 37 of file PythiaWrapper6_152.h.

9.52.2.7 int kchg[4][500]

Definition at line 51 of file PythiaWrapper6_152.h.

9.52.2.8 int kfdp[5][4000]

Definition at line 59 of file PythiaWrapper6_152.h.

9.52.2.9 int kfin[81][2]

Definition at line 70 of file PythiaWrapper6_152.h.

9.52.2.10 int kfpr[2][500]

Definition at line 90 of file PythiaWrapper6_152.h.

9.52.2.11 int mdcy[3][500]

Definition at line 57 of file PythiaWrapper6_152.h.

9.52.2.12 int mdme[2][4000]

Definition at line 57 of file PythiaWrapper6_152.h.

9.52.2.13 int mint[400]

Definition at line 84 of file PythiaWrapper6_152.h.

9.52.2.14 int mrpy[6]

Definition at line 64 of file PythiaWrapper6_152.h.

9.52.2.15 int msel

Definition at line 70 of file PythiaWrapper6_152.h.

9.52.2.16 int mselpd

Definition at line 70 of file PythiaWrapper6_152.h.

9.52.2.17 int msti[200]

Definition at line 78 of file PythiaWrapper6_152.h.

9.52.2.18 int mstj[200]

Definition at line 45 of file PythiaWrapper6_152.h.

9.52.2.19 int mstp[200]

Definition at line 76 of file PythiaWrapper6_152.h.

9.52.2.20 int mstu[200]

Definition at line 43 of file PythiaWrapper6_152.h.

9.52.2.21 int msub[500]

Definition at line 70 of file PythiaWrapper6_152.h.

9.52.2.22 int n

Definition at line 37 of file PythiaWrapper6_152.h.

9.52.2.23 int ngen[3][501]

Definition at line 97 of file PythiaWrapper6_152.h.

9.52.2.24 int ngenpd

Definition at line 97 of file PythiaWrapper6_152.h.

9.52.2.25 int npad

Definition at line 37 of file PythiaWrapper6_152.h.

9.52.2.26 double p[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_152.h.

9.52.2.27 double parf[2000]

Definition at line 52 of file PythiaWrapper6_152.h.

9.52.2.28 double pari[200]

Definition at line 79 of file PythiaWrapper6_152.h.

9.52.2.29 double parj[200]

Definition at line 46 of file PythiaWrapper6_152.h.

9.52.2.30 double parp[200]

Definition at line 77 of file PythiaWrapper6_152.h.

9.52.2.31 double paru[200]

Definition at line 44 of file PythiaWrapper6_152.h.

9.52.2.32 double pmas[4][500]

Definition at line 52 of file PythiaWrapper6_152.h.

9.52.2.33 struct { ... } pydat1_

9.52.2.34 struct { ... } pydat2_

9.52.2.35 struct { ... } pydat3_

9.52.2.36 struct { ... } pydatr_

9.52.2.37 struct { ... } pyint1_

9.52.2.38 struct { ... } pyint2_

9.52.2.39 struct { ... } pyint5_

9.52.2.40 struct { ... } pyjets_

9.52.2.41 struct { ... } pypars_

9.52.2.42 struct { ... } pysubs_

9.52.2.43 double rrpy[100]

Definition at line 65 of file PythiaWrapper6_152.h.

9.52.2.44 double v[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_152.h.

9.52.2.45 double vckm[4][4]

Definition at line 52 of file PythiaWrapper6_152.h.

9.52.2.46 double vint[400]

Definition at line 85 of file PythiaWrapper6_152.h.

9.52.2.47 double xsec[3][501]

Definition at line 98 of file PythiaWrapper6_152.h.

9.53 PythiaWrapper6_152_WIN32.h File Reference

9.54 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 **pystat** pystat_
- #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 **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_**

9.54.1 Define Documentation

9.54.1.1 #define initpydata initpydata_

Definition at line 30 of file PythiaWrapper6_2.h.

9.54.1.2 #define pydat1 pydat1_

Definition at line 52 of file PythiaWrapper6_2.h.

9.54.1.3 #define pydat2 pydat2_

Definition at line 60 of file PythiaWrapper6_2.h.

9.54.1.4 #define pydat3 pydat3_

Definition at line 69 of file PythiaWrapper6_2.h.

9.54.1.5 #define pydata pydata_

9.54.1.6 #define pydatr pydatr_

Definition at line 77 of file PythiaWrapper6_2.h.

9.54.1.7 #define pyevnt pyevnt_

Definition at line 129 of file PythiaWrapper6_2.h.

9.54.1.8 #define pyhepc pyhepc_

Definition at line 125 of file PythiaWrapper6_2.h.

9.54.1.9 #define pyinit pyinit_

Definition at line 126 of file PythiaWrapper6_2.h.

9.54.1.10 #define pyint1 pyint1_

Definition at line 103 of file PythiaWrapper6_2.h.

9.54.1.11 #define pyint2 pyint2_

Definition at line 112 of file PythiaWrapper6_2.h.

9.54.1.12 #define pyint5 pyint5_

Definition at line 120 of file PythiaWrapper6_2.h.

9.54.1.13 #define pyjets pyjets_

Definition at line 42 of file PythiaWrapper6_2.h.

9.54.1.14 #define pylist pylist_

Definition at line 127 of file PythiaWrapper6_2.h.

9.54.1.15 #define pypars pypars_

Definition at line 95 of file PythiaWrapper6_2.h.

9.54.1.16 #define pystat pystat_

Definition at line 128 of file PythiaWrapper6_2.h.

9.54.1.17 #define pysubs pysubs_

Definition at line 85 of file PythiaWrapper6_2.h.

9.54.1.18 #define upevnt upevnt_

Definition at line 131 of file PythiaWrapper6_2.h.

9.54.1.19 #define upinit upinit_

Definition at line 130 of file PythiaWrapper6_2.h.

9.54.2 Function Documentation

9.54.2.1 void initpydata (void)

9.54.3 Variable Documentation

9.54.3.1 double brat[8000]

Definition at line 65 of file PythiaWrapper6_2.h.

9.54.3.2 double ckin[200]

Definition at line 82 of file PythiaWrapper6_2.h.

9.54.3.3 double coef[20][500]

Definition at line 108 of file PythiaWrapper6_2.h.

9.54.3.4 int icol[2][4][40]

Definition at line 109 of file PythiaWrapper6_2.h.

9.54.3.5 int iset[500]

Definition at line 107 of file PythiaWrapper6_2.h.

9.54.3.6 int k[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_2.h.

9.54.3.7 int kchg[4][500]

Definition at line 56 of file PythiaWrapper6_2.h.

9.54.3.8 int kfdp[5][8000]

Definition at line 66 of file PythiaWrapper6_2.h.

9.54.3.9 int kfin[81][2]

Definition at line 81 of file PythiaWrapper6_2.h.

9.54.3.10 int kfpr[2][500]

Definition at line 107 of file PythiaWrapper6_2.h.

9.54.3.11 int mdcy[3][500]

Definition at line 64 of file PythiaWrapper6_2.h.

9.54.3.12 int mdme[2][8000]

Definition at line 64 of file PythiaWrapper6_2.h.

9.54.3.13 int mint[400]

Definition at line 99 of file PythiaWrapper6_2.h.

9.54.3.14 int mrpy[6]

Definition at line 73 of file PythiaWrapper6_2.h.

9.54.3.15 int msel

Definition at line 81 of file PythiaWrapper6_2.h.

9.54.3.16 int mselpd

Definition at line 81 of file PythiaWrapper6_2.h.

9.54.3.17 int msti[200]

Definition at line 91 of file PythiaWrapper6_2.h.

9.54.3.18 int mstj[200]

Definition at line 48 of file PythiaWrapper6_2.h.

9.54.3.19 int mstp[200]

Definition at line 89 of file PythiaWrapper6_2.h.

9.54.3.20 int mstu[200]

Definition at line 46 of file PythiaWrapper6_2.h.

9.54.3.21 int msub[500]

Definition at line 81 of file PythiaWrapper6_2.h.

9.54.3.22 int n

Definition at line 38 of file PythiaWrapper6_2.h.

9.54.3.23 int ngen[3][501]

Definition at line 116 of file PythiaWrapper6_2.h.

9.54.3.24 int ngenpd

Definition at line 116 of file PythiaWrapper6_2.h.

9.54.3.25 int npad

Definition at line 38 of file PythiaWrapper6_2.h.

9.54.3.26 double p[5][pyjets_maxn]

Definition at line 39 of file PythiaWrapper6_2.h.

9.54.3.27 double parf[2000]

Definition at line 57 of file PythiaWrapper6_2.h.

9.54.3.28 double pari[200]

Definition at line 92 of file PythiaWrapper6_2.h.

9.54.3.29 double parj[200]

Definition at line 49 of file PythiaWrapper6_2.h.

9.54.3.30 double parp[200]

Definition at line 90 of file PythiaWrapper6_2.h.

9.54.3.31 double paru[200]

Definition at line 47 of file PythiaWrapper6_2.h.

9.54.3.32 double pmas[4][500]

Definition at line 57 of file PythiaWrapper6_2.h.

9.54.3.33 struct { ... } pydat1_

9.54.3.34 struct { ... } pydat2_

9.54.3.35 struct { ... } pydat3_

9.54.3.36 struct { ... } pydatr_

9.54.3.37 struct { ... } pyint1_

9.54.3.38 struct { ... } pyint2_

9.54.3.39 struct { ... } pyint5_

9.54.3.40 struct { ... } pyjets_

9.54.3.41 const int pyjets_maxn = 4000

Definition at line 35 of file PythiaWrapper6_2.h.

9.54.3.42 struct { ... } pypars_

9.54.3.43 struct { ... } pysubs_

9.54.3.44 double rrpy[100]

Definition at line 74 of file PythiaWrapper6_2.h.

9.54.3.45 double v[5][pyjets_maxn]

Definition at line 39 of file PythiaWrapper6_2.h.

9.54.3.46 double vckm[4][4]

Definition at line 57 of file PythiaWrapper6_2.h.

9.54.3.47 double vint[400]

Definition at line 100 of file PythiaWrapper6_2.h.

9.54.3.48 double xsec[3][501]

Definition at line 117 of file PythiaWrapper6_2.h.

9.55 PythiaWrapper6_2_WIN32.h File Reference

9.56 SearchVector.cc File Reference

```
#include "HepMC/SearchVector.h"
```

Namespaces

- namespace **HepMC**

Functions

- bool **HepMC::not_in_vector** (std::vector< GenParticle * > *v, GenParticle *p)
- std::vector< **HepMC::GenParticle** * >::iterator **HepMC::already_in_vector** (std::vector< GenParticle * > *v, GenParticle *p)

  *returns true if **GenParticle** (p. 77) is in the vector*

9.57 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< **HepMC::GenParticle** * > *, GenParticle *)

9.58 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. *38) is a simple representation of a physics 4 vector.*
- class **HepMC::ThreeVector**
  ThreeVector *(*p. *216) is a simple representation of a position or displacement 3 vector.*

9.59 TempParticleMap.h File Reference

```
#include <map>
```

Namespaces

- namespace **HepMC**

Classes

- class **HepMC::TempParticleMap**

  TempParticleMap *(p. 213) is a temporary GenParticle* container used during input.*

9.60 testHepMCIteration.h File Reference

Classes

- class **IsFinalState**
  example class

Functions

- bool **IsPhoton** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle particle is a photon with pT > 10 GeV
- bool **IsWBoson** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle is a W+/W-

9.60.1 Function Documentation

9.60.1.1 bool IsPhoton (const HepMC::GenParticle ∗ p)

returns true if the GenParticle particle is a photon with pT > 10 GeV

Definition at line 10 of file testHepMCIteration.h.

References p.

9.60.1.2 bool IsWBoson (const HepMC::GenParticle ∗ p)

returns true if the GenParticle is a W+/W-

Definition at line 17 of file testHepMCIteration.h.

References p.

9.61 testPrintBug.cc File Reference

```
#include <fstream>
#include "HepMC/GenEvent.h"
#include "HepMC/SimpleVector.h"
```

Functions

- int **main** (int argc, char *argv[])

9.61.1 Function Documentation

9.61.1.1 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().

9.62 testSimpleVector.cc File Reference

```
#include <iostream>
```

```
#include "HepMC/SimpleVector.h"
```

Functions

- int **main** ()

9.62.1 Function Documentation

9.62.1.1 int main ()

Definition at line 8 of file testSimpleVector.cc.

References HepMC::FourVector::eta(), HepMC::FourVector::m(), HepMC::FourVector::m2(), HepMC::ThreeVector::mag(), HepMC::ThreeVector::perp(), HepMC::ThreeVector::perp2(), HepMC::ThreeVector::phi(), HepMC::FourVector::pseudoRapidity(), HepMC::ThreeVector::r(), 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::ThreeVector::theta(), v, HepMC::FourVector::x(), HepMC::ThreeVector::x(), HepMC::FourVector::y(), HepMC::ThreeVector::y(), HepMC::FourVector::z(), and HepMC::ThreeVector::z().

9.63 VectorConversion.h File Reference

```
#include "HepMC/SimpleVector.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Functions

- **CLHEP::HepLorentzVector SVtoLV** (const **HepMC::ThreeVector** &**v**)
- **CLHEP::HepLorentzVector SVtoLV** (const **HepMC::FourVector** &**v**)

9.63.1 Function Documentation

9.63.1.1 CLHEP::HepLorentzVector SVtoLV (const HepMC::FourVector & *v*)

Definition at line 15 of file VectorConversion.h.

References v.

9.63.1.2 CLHEP::HepLorentzVector SVtoLV (const HepMC::ThreeVector & *v*)

Examples:

example_BuildEventFromScratch.cc.

Definition at line 12 of file VectorConversion.h.

References v.

Referenced by main().

9.64 Version.h File Reference

```
#include <string>
#include <iostream>
```

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*

9.65 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

```
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
7 // To Compile: go to the HepMC directory and type:
8 // gmake examples/example_BuildEventFromScratch.exe
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     22    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     //                        p7
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     //
72     // create vertex 1 and vertex 2, together with their inparticles
73     GenVertex* v1 = new GenVertex();
74     evt->add_vertex( v1 );
75     v1->add_particle_in( new GenParticle( HepLorentzVector(0,0,7000,7000),
76                                       2212, 3 ) );
77     GenVertex* v2 = new GenVertex();
78     evt->add_vertex( v2 );
79     v2->add_particle_in( new GenParticle( HepLorentzVector(0,0,-7000,7000),
80                                       2212, 3 ) );
81     //
82     // create the outgoing particles of v1 and v2
83     GenParticle* p3 =
84         new GenParticle( HepLorentzVector(.750,-1.569,32.191,32.238), 1, 3 );
85     v1->add_particle_out( p3 );
86     GenParticle* p4 =
87         new GenParticle( HepLorentzVector(-3.047,-19.,-54.629,57.920), -2, 3 );
88     v2->add_particle_out( p4 );
89     //
90     // create v3
91     GenVertex* v3 = new GenVertex();
92     evt->add_vertex( v3 );
93     v3->add_particle_in( p3 );
94     v3->add_particle_in( p4 );
95     v3->add_particle_out(
96         new GenParticle( HepLorentzVector(-3.813,0.113,-1.833,4.233 ), 22, 1 )
97         );
98     GenParticle* p5 =
99         new GenParticle( HepLorentzVector(1.517,-20.68,-20.605,85.925), -24,3);
100     v3->add_particle_out( p5 );
101     //
102     // create v4
103     GenVertex* v4 = new GenVertex(HepLorentzVector(0.12,-0.3,0.05,0.004));
104     evt->add_vertex( v4 );
105     v4->add_particle_in( p5 );
106     v4->add_particle_out(
107         new GenParticle( HepLorentzVector(-2.445,28.816,6.082,29.552), 1,1 )
108         );
109     v4->add_particle_out(
110         new GenParticle( HepLorentzVector(3.962,-49.498,-26.687,56.373), -2,1 )
111         );
112     //
```

```
113     // tell the event which vertex is the signal process vertex
114     evt->set_signal_process_vertex( v3 );
115     // the event is complete, we now print it out to the screen
116     evt->print();
117
118     // example conversion back to Lorentz vector
119     // add all outgoing momenta
120     std::cout << std::endl;
121     std::cout << " Add output momenta " << std::endl;
122     HepLorentzVector sum;
123     for ( GenEvent::particle_const_iterator p = evt->particles_begin();
124               p != evt->particles_end(); ++p ){
125         if( (*p)->status() == 1 ) {
126             sum += SVtoLV( (*p)->momentum() );
127             (*p)->print();
128         }
129     }
130     std::cout << "Vector Sum: " << sum << std::endl;
131
132     // now clean-up by deleteing all objects from memory
133     //
134     // deleting the event deletes all contained vertices, and all particles
135     // contained in those vertices
136     delete evt;
137
138     // delete all particle data objects in the particle data table pdt
139     pdt.delete_all();
140
141     return 0;
142 }
```

10.2 example_EventSelection.cc

```
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 pT > 25 GeV pass the selection and are
6 // written to "example_EventSelection.dat"
8 // To Compile: go to the HepMC directory and type:
9 // gmake examples/example_EventSelection.exe
10 //
11
12 #include "HepMC/IO_Ascii.h"
13 #include "HepMC/GenEvent.h"
14
16
20 class IsGoodEvent {
21 public:
23     bool operator()( const HepMC::GenEvent* evt ) {
24         for ( HepMC::GenEvent::particle_const_iterator p
25                   = evt->particles_begin(); p != evt->particles_end(); ++p ){
26             if ( (*p)->pdg_id() == 22 && (*p)->momentum().perp() > 25. ) {
27                 //std::cout << "Event " << evt->event_number()
28                 //     << " is a good event." << std::endl;
29                 //(*p)->print();
30                 return 1;
31             }
32         }
33         return 0;
34     }
35 };
36
37 int main() {
38     // declare an input strategy to read the data produced with the
39     // example_MyPythia
40     { // begin scope of ascii_in and ascii_out
41         HepMC::IO_Ascii ascii_in("example_MyPythia.dat",std::ios::in);
42         // declare another IO_Ascii for writing out the good events
43         HepMC::IO_Ascii ascii_out("example_EventSelection.dat",std::ios::out);
44         // declare an instance of the event selection predicate
45         IsGoodEvent is_good_event;
46         //.....................................EVENT LOOP
47         int icount=0;
48         int num_good_events=0;
49         HepMC::GenEvent* evt = ascii_in.read_next_event();
50         while ( evt ) {
51             icount++;
52             if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
53                                           << " its # " << evt->event_number()
54                                           << std::endl;
55             if ( is_good_event(evt) ) {
56                 ascii_out << evt;
57                 ++num_good_events;
58             }
59             delete evt;
60             ascii_in >> evt;
61         }
62         //.....................................PRINT RESULT
63         std::cout << num_good_events << " out of " << icount
64                   << " processed events passed the cuts. Finished." << std::endl;
65     } // end scope of ascii_in and ascii_out
66     return 0;
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.
16
17 #include <iostream>
18 #include "HepMC/HerwigWrapper.h"
19 #include "HepMC/IO_HERWIG.h"
20 #include "HepMC/GenEvent.h"
21 #include "HepMC/HEPEVT_Wrapper.h"
22
23 int main() {
24     //
25     //.....................................HEPEVT
26     // Herwig 6.4 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     //.....................................INITIALIZATIONS
34
35     hwproc.PBEAM1 = 7000.; // energy of beam1
36     hwproc.PBEAM2 = 7000.; // energy of beam2
37     // 1610 = gg->H--> WW, 1706 = qq-->ttbar, 2510 = ttH -> ttWW
38     hwproc.IPROC = 1706; // qq -> ttbar production
39     hwproc.MAXEV = 100; // number of events
40     // tell it what the beam particles are:
41     for ( unsigned int i = 0; i < 8; ++i ) {
42         hwbmch.PART1[i] = (i < 1) ? 'P' : ' ';
43         hwbmch.PART2[i] = (i < 1) ? 'P' : ' ';
44     }
45     hwigin();    // INITIALISE OTHER COMMON BLOCKS
46     hwevnt.MAXPR = 1; // number of events to print
47     hwuinc(); // compute parameter-dependent constants
48     hweini(); // initialise elementary process
49
50     //.....................................HepMC INITIALIZATIONS
51     //
52     // Instantiate an IO strategy for reading from HEPEVT.
53     HepMC::IO_HERWIG hepevtio;
54     //
55     //.....................................EVENT LOOP
56     for ( int i = 1; i <= hwproc.MAXEV; i++ ) {
57         if ( i%50==1 ) std::cout << "Processing Event Number "
58                                  << i << std::endl;
59         // initialise event
60         hwuine();
61         // generate hard subprocess
62         hwepro();
63         // generate parton cascades
64         hwbgen();
65         // do heavy object decays
66         hwdhob();
67         // do cluster formation
68         hwcfor();
69         // do cluster decays
70         hwcdec();
71         // do unstable particle decays
72         hwdhad();
73         // do heavy flavour hadron decays
74         hwdhvy();
75         // add soft underlying event if needed
```

```
76         hwmevt();
77         // finish event
78         hwufne();
79         HepMC::GenEvent* evt = hepevtio.read_next_event();
80         // add some information to the event
81         evt->set_event_number(i);
82         evt->set_signal_process_id(20);
83         if (i<=hwevnt.MAXPR) {
84             std::cout << "\n\n This is the FIXED version of HEPEVT as "
85                       << "coded in IO_HERWIG " << std::endl;
86             HepMC::HEPEVT_Wrapper::print_hepevt();
87             evt->print();
88         }
89
90         // we also need to delete the created event from memory
91         delete evt;
92     }
93     //.....................................TERMINATION
94     hwefin();
95
96     return 0;
97 }
```

10.4 example_MyPythia.cc

```
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 and then output to file in
8 // ascii format using the IO_Ascii strategy.
20
21 #include <iostream>
22 #include "HepMC/PythiaWrapper.h"
23 #include "HepMC/IO_HEPEVT.h"
24 #include "HepMC/IO_Ascii.h"
25 #include "HepMC/IO_ExtendedAscii.h"
26 #include "HepMC/GenEvent.h"
27 #include "PythiaHelper.h"
28
29 int main() {
30     //
31     //.....................................HEPEVT
32     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
33     //  numbers. We need to explicitly pass this information to the
34     //  HEPEVT_Wrapper.
35     //
36     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
37     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
38     //
39     //.....................................PYTHIA INITIALIZATIONS
40     initPythia();
41
42     //.....................................HepMC INITIALIZATIONS
43     //
44     // Instantiate an IO strategy for reading from HEPEVT.
45     HepMC::IO_HEPEVT hepevtio;
46     //
47     { // begin scope of ascii_io
48         // Instantiate an IO strategy to write the data to file
49         HepMC::IO_Ascii ascii_io("example_MyPythia.dat",std::ios::out);
50         // declare an IO_ExtendedAscii for output
51         HepMC::IO_ExtendedAscii xout("example_MyPythia.exdat",std::ios::out);
52         //
53         //.....................................EVENT LOOP
54         for ( int i = 1; i <= 100; i++ ) {
55             if ( i%50==1 ) std::cout << "Processing Event Number "
56                                      << i << std::endl;
57             call_pyevnt();      // generate one event with Pythia
58             // pythia pyhepc routine converts common PYJETS in common HEPEVT
59             call_pyhepc( 1 );
60             HepMC::GenEvent* evt = hepevtio.read_next_event();
61             // add some information to the event
62             evt->set_event_number(i);
63             evt->set_signal_process_id(20);
64             // set number of multi parton interactions
65             evt->set_mpi( pypars.msti[31-1] );
66             // write the event out to the ascii files
67             ascii_io << evt;
68             xout << evt;
69             // we also need to delete the created event from memory
70             delete evt;
71         }
72         //.....................................TERMINATION
73         // write out some information from Pythia to the screen
74         call_pystat( 1 );
75     } // end scope of ascii_io
76
```

```
77     return 0;
78 }
79
80
81
```

10.5 example_MyPythiaOnlyToHepMC.cc

```
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         // set number of multi parton interactions
50         evt->set_mpi( pypars.msti[31-1] );
51         //
52         //.....................USER WOULD PROCESS EVENT HERE
53         //
54         // we also need to delete the created event from memory
55         delete evt;
56     }
57     //.....................................TERMINATION
58     // write out some information from Pythia to the screen
59     call_pystat( 1 );
60
61     return 0;
62 }
63
64
65
```

10.6 example_MyPythiaRead.cc

```
1
2 // garren@fnal.gov, January 2007
3 // This example is an extension of example_MyPythia.cc
4 //
5 // generate events with Pythia, write a file, and read the resulting output
6 // Notice that we use scope to explicitly close the ouput files.
7 // The two output files should be the same size, but because particles are
8 // saved as sets within a vertex, they will be written in arbitrary order.
10 // To Compile: go to the HepMC directory and type:
11 // gmake examples/example_MyPythiaRead.exe
12 //
13 // In this example the precision and number of entries for the HEPEVT
14 // fortran common block are explicitly defined to correspond to those
15 // used in the Pythia version of the HEPEVT common block.
16 //
17 // If you get funny output from HEPEVT in your own code, probably you have
18 // set these values incorrectly!
19 //
20
21 #include <iostream>
22 #include "HepMC/PythiaWrapper.h"
23 #include "HepMC/IO_HEPEVT.h"
24 #include "HepMC/IO_Ascii.h"
25 #include "HepMC/GenEvent.h"
26 #include "PythiaHelper.h"
27
28 int main() {
29     //
30     //.....................................HEPEVT
31     // Pythia 6.3 uses HEPEVT with 4000 entries and 8-byte floating point
32     //  numbers. We need to explicitly pass this information to the
33     //  HEPEVT_Wrapper.
34     //
35     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
36     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
37     //
38     //.....................................PYTHIA INITIALIZATIONS
39     initPythia();
40
41     //.....................................HepMC INITIALIZATIONS
42     //
43     // Instantiate an IO strategy for reading from HEPEVT.
44     HepMC::IO_HEPEVT hepevtio;
45     //
46     //.....................................define the output scope
47     {
48         // Instantial an IO strategy to write the data to file - it uses the
49         //  same ParticleDataTable
50         HepMC::IO_Ascii ascii_io("example_MyPythiaRead.dat",std::ios::out);
51         //
52         //.....................................EVENT LOOP
53         for ( int i = 1; i <= 100; i++ ) {
54             if ( i%50==1 ) std::cout << "Processing Event Number "
55                                      << i << std::endl;
56             call_pyevnt();      // generate one event with Pythia
57             // pythia pyhepc routine converts common PYJETS in common HEPEVT
58             call_pyhepc( 1 );
59             HepMC::GenEvent* evt = hepevtio.read_next_event();
60             // add some information to the event
61             evt->set_event_number(i);
62             evt->set_signal_process_id(20);
63             // write the event out to the ascii file
64             ascii_io << evt;
65             // we also need to delete the created event from memory
66             delete evt;
```

```
67         }
68         //.....................................TERMINATION
69         // write out some information from Pythia to the screen
70         call_pystat( 1 );
71     }  // ascii_io destructor is called here
72     //
73     //.....................................define an input scope
74     {
75         // now read the file we wrote
76         HepMC::IO_Ascii ascii_in("example_MyPythiaRead.dat",std::ios::in);
77         HepMC::IO_Ascii ascii_io2("example_MyPythiaRead2.dat",std::ios::out);
78         int icount=0;
79         HepMC::GenEvent* evt = ascii_in.read_next_event();
80         while ( evt ) {
81             icount++;
82             if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
83                                           << " its # " << evt->event_number()
84                                           << std::endl;
85             // write the event out to the ascii file
86             ascii_io2 << evt;
87             delete evt;
88             ascii_in >> evt;
89         }
90         //.....................................PRINT RESULT
91         std::cout << icount << " events found. Finished." << std::endl;
92     } // ascii_io2 and ascii_in destructors are called here
93
94     return 0;
95 }
96
97
98
```

10.7 example_MyPythiaWithEventSelection.cc

```
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 and then a very simple event
8 // selection is performed.
10 // To Compile: go to the HepMC directory and type:
11 // gmake examples/example_MyPythiaWithEventSelection.exe
12 //
13 // See comments in examples/example_MyPythia.cxx regarding the HEPEVT wrapper.
14 //
15
16 #include <iostream>
17 #include "HepMC/PythiaWrapper.h"
18 #include "HepMC/IO_HEPEVT.h"
19 #include "HepMC/GenEvent.h"
20 #include "PythiaHelper.h"
21
22
24
28 class IsGoodEventMyPythia {
29 public:
31     bool operator()( const HepMC::GenEvent* evt ) {
32         for ( HepMC::GenEvent::particle_const_iterator p
33                   = evt->particles_begin(); p != evt->particles_end(); ++p ){
34             if ( (*p)->pdg_id() == 22 && (*p)->momentum().perp() > 25. ) {
35                 //std::cout << "Event " << evt->event_number()
36                 //     << " is a good event." << std::endl;
37                 //(*p)->print();
38                 return 1;
39             }
40         }
41         return 0;
42     }
43 };
44
45 int main() {
46     //
47     //.....................................HEPEVT
48     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
49     //  numbers. We need to explicitly pass this information to the
50     //  HEPEVT_Wrapper.
51     //
52     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
53     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
54     //
55     //.....................................PYTHIA INITIALIZATIONS
56     initPythia();
57     //
58     //.....................................HepMC INITIALIZATIONS
59     // Instantiate an IO strategy for reading from HEPEVT.
60     HepMC::IO_HEPEVT hepevtio;
61     // declare an instance of the event selection predicate
62     IsGoodEventMyPythia is_good_event;
63     //.....................................EVENT LOOP
64     int icount=0;
65     int num_good_events=0;
66     for ( int i = 1; i <= 100; i++ ) {
67         icount++;
68         if ( i%50==1 ) std::cout << "Processing Event Number "
69                                  << i << std::endl;
70         call_pyevnt(); // generate one event with Pythia
71         // pythia pyhepc routine convert common PYJETS in common HEPEVT
```

```
72         call_pyhepc( 1 );
73         HepMC::GenEvent* evt = hepevtio.read_next_event();
74         // set number of multi parton interactions
75         evt->set_mpi( pypars.msti[31-1] );
76         // do event selection
77         if ( is_good_event(evt) ) ++num_good_events;
78         // we also need to delete the created event from memory
79         delete evt;
80     }
81     //.....................................TERMINATION
82     // write out some information from Pythia to the screen
83     call_pystat( 1 );
84     //.....................................PRINT RESULTS
85     std::cout << num_good_events << " out of " << icount
86               << " processed events passed the cuts. Finished." << std::endl;
87     return 0;
88 }
89
90
91
```

10.8 example_PythiaParticle.cc

```
1
2 // garren@fnal.gov, July 2006
3 // example of generating events with Pythia
4 // using HepMC/PythiaWrapper.h
5 // Events are read into the HepMC event record from the FORTRAN HEPEVT
6 // common block using the IO_HEPEVT strategy and then output to file in
7 // ascii format using the IO_AsciiParticles strategy.
8 //
9 // This is identical to example_MyPythia.cc except that it uses IO_AsciiParticles.
11 // To Compile: go to the examples directory and type:
12 // gmake example_PythiaParticle.exe
13 //
14 // In this example the precision and number of entries for the HEPEVT
15 // fortran common block are explicitly defined to correspond to those
16 // used in the Pythia version of the HEPEVT common block.
17 //
18 // If you get funny output from HEPEVT in your own code, probably you have
19 // set these values incorrectly!
20 //
21
22 #include <iostream>
23 #include "HepMC/PythiaWrapper.h"
24 #include "HepMC/IO_HEPEVT.h"
25 #include "HepMC/IO_AsciiParticles.h"
26 #include "HepMC/GenEvent.h"
27 #include "PythiaHelper.h"
28
29 int main() {
30     //
31     //.....................................HEPEVT
32     // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
33     //  numbers. We need to explicitly pass this information to the
34     //  HEPEVT_Wrapper.
35     //
36     HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
37     HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
38     //
39     //.....................................PYTHIA INITIALIZATIONS
40     initPythia();
41
42     //.....................................HepMC INITIALIZATIONS
43     //
44     // Instantiate an IO strategy for reading from HEPEVT.
45     HepMC::IO_HEPEVT hepevtio;
46     //
47     { // begin scope of ascii_io
48         // Instantiate an IO strategy to write the data to file
49         HepMC::IO_AsciiParticles ascii_io("example_PythiaParticle.dat",std::ios::out);
50         //
51         //.....................................EVENT LOOP
52         for ( int i = 1; i <= 100; i++ ) {
53             if ( i%50==1 ) std::cout << "Processing Event Number "
54                                      << i << std::endl;
55             call_pyevnt();      // generate one event with Pythia
56             // pythia pyhepc routine converts common PYJETS in common HEPEVT
57             call_pyhepc( 1 );
58             HepMC::GenEvent* evt = hepevtio.read_next_event();
59             // add some information to the event
60             evt->set_event_number(i);
61             evt->set_signal_process_id(20);
62             // write the event out to the ascii file
63             ascii_io << evt;
64             // we also need to delete the created event from memory
65             delete evt;
66         }
```

```
67         //.....................................TERMINATION
68         // write out some information from Pythia to the screen
69         call_pystat( 1 );
70     } // end scope of ascii_io
71
72     return 0;
73 }
74
75
76
```

10.9 example_UsingIterators.cc

```
1
2 // Matt.Dobbs@Cern.CH, Feb 2000
3 // This example shows low 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_Ascii.h"
10 #include "HepMC/GenEvent.h"
11 #include <math.h>
12 #include <algorithm>
13 #include <list>
14
16
20 class IsPhoton {
21 public:
23     bool operator()( const HepMC::GenParticle* p ) {
24         if ( p->pdg_id() == 22
25              && p->momentum().perp() > 10. ) return 1;
26         return 0;
27     }
28 };
29
31
34 class IsW_Boson {
35 public:
37     bool operator()( const HepMC::GenParticle* p ) {
38         if ( abs(p->pdg_id()) == 24 ) return 1;
39         return 0;
40     }
41 };
42
44
47 class IsFinalState {
48 public:
50     bool operator()( const HepMC::GenParticle* p ) {
51         if ( !p->end_vertex() && p->status()==1 ) return 1;
52         return 0;
53     }
54 };
55
56 int main() {
57     { // begin scope of ascii_in
58         // an event has been prepared in advance for this example, read it
59         // into memory using the IO_Ascii input strategy
60         HepMC::IO_Ascii ascii_in("example_UsingIterators.txt",std::ios::in);
61         if ( ascii_in.rdstate() == std::ios::failbit ) {
62             std::cerr << "ERROR input file example_UsingIterators.txt is needed "
63                       << "and does not exist. "
64                       << "\n Look for it in HepMC/examples, Exit." << std::endl;
65             return 1;
66         }
67
68         HepMC::GenEvent* evt = ascii_in.read_next_event();
69
70         // if you wish to have a look at the event, then use evt->print();
71
72         // use GenEvent::vertex_iterator to fill a list of all
73         // vertices in the event
74         std::list<HepMC::GenVertex*> allvertices;
75         for ( HepMC::GenEvent::vertex_iterator v = evt->vertices_begin();
76               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 IsFinalState predicate
87        IsFinalState 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), IsFinalState() );
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 }
```