PYTHIA 8.1
Introduction and Tutorial

Torbjörn Sjöstrand
CERN/PH and Department of Theoretical Physics, Lund University
PYTHIA 6 status

PYTHIA has its roots in JETSET, begun in 1978 → almost 30 years. PYTHIA 6 still being (slightly) developed and (fully) maintained:

- multiple interactions and underlying event, with
- transverse-momentum-ordered showers
- SUSY interfaces (SLHA) and simulation
- regular bug fixes and minor improvements
- moved to CEDAR HepForge (code management, bugtracking)

Currently PYTHIA 6.413:

- 75,000 lines of code (including comments/blanks)
- 580 page PYTHIA 6.4 Physics and Manual
  T. Sjöstrand, S. Mrenna and P. Skands,
- + update notes, sample main programs, etc.

...but

- only add, never subtract
  ⇒ has become bloated and unmanageable
- is in Fortran 77, so not understood by young people
PYTHIA 8: plans and reality

Tentative schedule (spring 2003):

<table>
<thead>
<tr>
<th>time</th>
<th>date</th>
<th>processes</th>
<th>final states</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 Sept. 2004</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>1</td>
<td>1 Sept. 2005</td>
<td>LHA-style input</td>
<td>incomplete draft</td>
</tr>
<tr>
<td>2</td>
<td>1 Sept. 2006</td>
<td>a few processes</td>
<td>complete, buggy(?)</td>
</tr>
<tr>
<td>3</td>
<td>1 Sept. 2007</td>
<td>more processes</td>
<td>stable, debugged</td>
</tr>
</tbody>
</table>

Status: involuntary break ~6 months + Murphy’s law
⇒⇒⇒ currently ~ at year 2.5

PYTHIA 8.100 released on 20 October:

- Webpages revamped
  - Recent ⇐ PYTHIA 6.4
  - Present ⇐ PYTHIA 8.1
  - Future ⇐ loose plans

- A Brief Introduction to PYTHIA 8.1
  in arXiv:0710.3820
  submitted to CPC
PYTHIA 8 status

<table>
<thead>
<tr>
<th>task</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>administrative structure</td>
<td>operational; extensions planned</td>
</tr>
<tr>
<td>hard processes, internal</td>
<td>much of PYTHIA 6; SUSY &amp; TC &amp; more to do</td>
</tr>
<tr>
<td>resonance decays</td>
<td>much of PYTHIA 6; SUSY &amp; TC &amp; more to do</td>
</tr>
<tr>
<td>hard processes, external</td>
<td>interfaces to LHA F77, LHEF, PYTHIA 6</td>
</tr>
<tr>
<td>SUSY(+more) parameters</td>
<td>primitive SLHA2; more needed</td>
</tr>
<tr>
<td>initial-state showers</td>
<td>operational</td>
</tr>
<tr>
<td>final-state showers</td>
<td>operational</td>
</tr>
<tr>
<td>matching ME’s to showers</td>
<td>some exists; much more needed</td>
</tr>
<tr>
<td>multiple interactions</td>
<td>operational; extensions planned</td>
</tr>
<tr>
<td>beam remnants &amp; colour flow</td>
<td>operational; alternatives to come</td>
</tr>
<tr>
<td>parton densities</td>
<td>only 2 internal, but interface to LHAPDF</td>
</tr>
<tr>
<td>string fragmentation</td>
<td>operational; improvements planned</td>
</tr>
<tr>
<td>decays &amp; particle data</td>
<td>operational; may need updates</td>
</tr>
<tr>
<td>Bose-Einstein</td>
<td>operational; off by default (tuning)</td>
</tr>
<tr>
<td>analysis</td>
<td>some simple tools; may be enough</td>
</tr>
<tr>
<td>graphical user interface</td>
<td>operational; could be extended</td>
</tr>
<tr>
<td>tuning</td>
<td>major task for MCnet postdocs!</td>
</tr>
<tr>
<td>testing</td>
<td>major task for experimentalists!</td>
</tr>
</tbody>
</table>
Key differences between PYTHIA 6.4 and 8.1

Old features definitely removed include, among others:
- independent fragmentation
- mass-ordered showers

Features omitted so far include, among others:
- ep, \(\gamma p\) and \(\gamma \gamma\) beam configurations
- several processes, especially SUSY & Technicolor

New features, not found in 6.4:
- interleaved \(p_{\perp}\)-ordered MI + ISR + FSR evolution
- richer mix of underlying-event processes (\(\gamma, J/\psi, DY, \ldots\))
- possibility for two selected hard interactions in same event
- possibility to use one PDF set for hard process and another for rest
- elastic scattering with Coulomb term (optional)
- updated decay data

Preliminary plans for the future:
- rescattering in multiple interactions
- NLO and L-CKKW matching
Trying It Out

- Download `pythia8100.tgz` from
  
  http://www.thep.lu.se/~torbjorn/Pythia.html
- `tar xvfz pythia8100.tgz` to unzip and expand
- `cd pythia8100` to move to new directory
- `./configure` ... needed for external libraries + debug/shared
  (see README, libraries: HepMC, LHAPDF, PYTHIA 6)
- `make` will compile in ~ 3 minutes
  (for archive library, same amount extra for shared)
- The `htmldoc/pythia8100.pdf` file contains A Brief Introduction
- Open `htmldoc/Welcome.html` in a web browser for the full manual
- Install the `phpdoc/` directory on a webserver and open `phpdoc/Welcome.html` in a web browser for an interactive manual
- The `examples` subdirectory contains 30 sample main programs:
  standalone, link to libraries, semi-internal processes, ...
  (make mainNN and then ./mainNN.exe > outfile)
- A Worksheet (on the web pages) contains step-by-step instructions and exercises how to write and run a main program
PYTHIA 8 structure

The User (≈ Main Program)

Pythia

Info

Event process

Event event

ProcessLevel
- ProcessContainer
- PhaseSpace
- LHAinit, LHAevnt
- ResonanceDecays

PartonLevel
- TimeShower
- SpaceShower
- MultipleInteractions
- BeamRemnants

HadronLevel
- StringFragmentation
- MiniStringFrag…
- ParticleDecays
- BoseEinstein

BeamParticle

SigmaProcess, SigmaTotal

Vec4, Rndm, Hist, Settings, ParticleDataTable, ResonanceWidths, …
Example of a main program

// File: main01.cc. The charged multiplicity distribution at the LHC.
#include "Pythia.h"
using namespace Pythia8;
int main() {
    Pythia pythia;
    pythia.readString("HardQCD:all = on");
    pythia.readString("PhaseSpace:pTHatMin = 20.");
    pythia.init( 2212, 2212, 14000.);
    Hist mult("charged multiplicity", 100, -0.5, 799.5);
    // Begin event loop. Generate event. Skip if error. List first one.
    for (int iEvent = 0; iEvent < 100; ++iEvent) {
        if (!pythia.next()) continue;
        if (iEvent < 1) {pythia.info.list(); pythia.event.list();}
        // Find number of all final charged particles and fill histogram.
        int nCharged = 0;
        for (int i = 0; i < pythia.event.size(); ++i)
            if (pythia.event[i].isFinal() && pythia.event[i].isCharged())
                ++nCharged;
        mult.fill(nCharged);
    }
    pythia.statistics();
    cout << mult;
    return 0;
}
Initialization and generation commands

Standard in beginning:
- #include "Pythia.h"
- using namespace Pythia8;
- Pythia pythia;

Initialization by one of different forms:
- pythia.init( idA, idB, eA, eB) along $\pm z$ axis
- pythia.init( idA, idB, eCM) in c.m. frame
- pythia.init("filename") for Les Houches Event Files
- pythia.init() takes above kinds of input from “cards”
- pythia.init( LHAinit*, LHAevnt*) for Les Houches Accord
  returns false if failed (normally user setup mistake!)

Generation of next event by:
- pythia.next()
  with no arguments, but value false if failed (rare!)

At the end of the generation loop:
- pythia.statistics()
  provides some summary information
Settings and Particle Data

Can read in settings and particle data changes by
- `pythia.readString("command")`
- `pythia.readFile("filename")` with one `command` per line in file

**Settings** come in four kinds
- **Flags**: on/off switches, bool
  (on = yes = ok = true = 1, off = no = false = 0)
- **Modes**: enumerated options, int
- **Parms**: (short for parameters) continuum of values, double
- **Words**: characters (no blanks), string

and **command** is of form `task:property = value`, e.g.
- `PartonLevel:ISR = off` no initial-state radiation
- `SigmaProcess:alphaSorder = 0` freeze $\alpha_s$
- `TimeShower:pTmin = 1.0` cut off final-state radiation at 1 GeV

To access **particle data**, instead **command** should be of form
- `id:property = value` or `id:channel:property = value`, e.g.
  - `3122:mayDecay = no` do not allow $\Lambda^0$ to decay
  - `215:3:products = 211 111 111` to let $a_2^+ \rightarrow \pi^+\pi^0\pi^0$

**Note**: case-insensitive search/matching in databases!
Example of a “cards” file

! This file contains commands to be read in for a Pythia8 run.
! Lines not beginning with a letter or digit are comments.

! 1) Settings that could be used in a main program, if desired.
Main:idBeamA = 2212 ! first beam, p = 2212, pbar = -2212
Main:idBeamB = 2212 ! second beam, p = 2212, pbar = -2212
Main:eCM = 14000. ! CM energy of collision
Main:numberOfEvents = 1000 ! number of events to generate
Main:numberToList = 2 ! number of events to print
Main:timesToShow = 20 ! show how far along run is
Main:showChangedSettings = on ! print changed flags/modes/parameters
Main:showAllSettings = off ! print all flags/modes/parameters

! 2) Settings for the hard-process generation.
HiggsSM:gg2H = on ! Higgs production by gluon-gluon fusion
25:m0 = 123.5 ! Higgs mass
25:onMode = off ! switch off all Higgs decay channels
25:onIfMatch = 22 22 ! switch back on Higgs -> gamma gamma
SigmaProcess:alphaSvalue = 0.12 ! alpha_s(m_Z) in matrix elements

! 3) Settings for the subsequent event generation process.
SpaceShower:alphaSvalue = 0.13 ! alpha_s(m_Z) in initial-state radiation
MultipleInteractions:pT0Ref = 3.0 ! pT_0 regularization at reference energy
#PartonLevel:MI = off ! no multiple interactions
#PartonLevel:ISR = off ! no initial-state radiation
#PartonLevel:FSR = off ! no final-state radiation
#HadronLevel:Hadronize = off ! no hadronization
More on settings

Settings are stored in four separate maps (flags/modes/parms/words). For each setting, need to store

- **name**: of form `task:property`, e.g. `TimeShower:pTmin`
- **default value**
- **current value**
- **allowed range**: minimum/maximum on/off (not for flags).

Useful commands:

- `pythia.settings.listAll()`: complete list
- `pythia.settings.listChanged()`: only changed ones

*------- PYTHIA Flag + Mode + Parm + Word Settings (changes only) --------------------------*

<table>
<thead>
<tr>
<th>Name</th>
<th>Now</th>
<th>Default</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>HardQCD:all</td>
<td>on</td>
<td>off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Main:eCM</td>
<td>14000.000</td>
<td>1960.000</td>
<td>10.00000</td>
<td></td>
</tr>
<tr>
<td>Main:numberToList</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Main:showChangedParticleData</td>
<td>on</td>
<td>off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Main:timesToShow</td>
<td>20</td>
<td>50</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MultipleInteractions:pTmin</td>
<td>3.00000</td>
<td>0.20000</td>
<td>0.10000</td>
<td>10.00000</td>
</tr>
<tr>
<td>PhaseSpace:pTHatMin</td>
<td>50.00000</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>PromptPhoton:all</td>
<td>on</td>
<td>off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SpaceShower:pT0Ref</td>
<td>2.00000</td>
<td>2.20000</td>
<td>0.50000</td>
<td>10.00000</td>
</tr>
</tbody>
</table>

*------- End PYTHIA Flag + Mode + Parm + Word Settings ------------------------------------*
Online manual ⇒ Graphical User Interface

PYTHIA 8

Welcome to PYTHIA - The Lund Monte Carlo!

PYTHIA 8 is the successor to PYTHIA 6, rewritten from scratch in C++. With the release of PYTHIA 8.1 it now becomes the official "current" PYTHIA version, although PYTHIA 6.4 will be supported in parallel with it for some time to come. Specifically, the new version has not yet been enough tested and tuned for it to have reached the same level of reliability as the older one. This testing will only happen if people begin to work with the program, however, which is why we encourage a gradual transition to the new version, starting now. There are some new physics features in PYTHIA 8.1, that would make use of it more attractive, but also some topics still missing, where 6.4 would have to be used. Further, many obsolete features will not be carried over, so for some backwards compatibility studies again 6.4 would be the choice.

Documentation

On these webpages you will find the up-to-date manual for PYTHIA 8.1. Use the left-hand index to navigate this documentation of program elements, especially all possible program settings. All parameters are provided with sensible default values, however, so you need only change those of relevance to your particular study, such as choice of beams, processes and phase space cuts. The pages also contain a fairly extensive overview of all methods available to the user, e.g. to study the produced events. What is lacking on these webpages is an overview, on the one hand, and an in-depth physics description, on the other.

Example: timelike parton showers

The choice is not as unique. Here the factorization scale has been chosen as the maximum evolution scale. This would be the $p_T$ for a $2 \rightarrow 2$ process, supplemented by mass terms for massive outgoing particles. Some small amount of freedom is offered by

\[ \text{TimeShower: pT\text{maxFudge}} = 1.0 \] (default = 1.0; minimum = 0.5; maximum = 2.0)

While the above rules would imply that $p_T\text{max} = p_T\text{factorization}$, pTmaxFudge introduced a multiplicative factor $f$ such that instead $p_T\text{max} = f \cdot p_T\text{factorization}$. Only applies to the hardest interaction in an event. It is strongly suggested that $f = 1$, but variations around this default can be useful to test this assumption.

The amount of QCD radiation in the shower is determined by

\[ \text{TimeShower: alphaSvalue} = 0.137 \] (default = 0.137; minimum = 0.06; maximum = 0.25)

The $alpha\_strong$ value at scale $M_Z^2$. The default value corresponds to a crude tuning to LEP data, to be improved.

The actual value is then regulated by the running to the scale $pT^2$, at which the shower evaluates $alpha\_strong$.

\[ \text{TimeShower: alphaSorder} \] (default = 1; minimum = 0; maximum = 2)

Order at which $alpha\_strong$ runs,

- 0: zeroth order, i.e. $alpha\_strong$ is kept fixed.
- 1: first order, which is the normal value.
- 2: second order. Since other parts of the code do not go to second order there is no strong reason to use this option, but there is also nothing wrong with it.
Manual Sections

Program Overview
- Frontpage
- Program Flow
- Settings Scheme
- Particle Data Scheme
- Program Files

Setup Run Tasks
- Save Settings
- Main-Program Settings
- Random-Number Seed
- PDF Selection
- Master Switches
- Process Selection
  - QCD
  - Electroweak
  - Onia
  - Top
  - Fourth Generation
  - Higgs
  - SUSY
  - New Gauge Bosons
  - Left-Right Symmetry
  - Leptoquark

- Compositeness
- Extra Dimensions
- A Second Hard Process
- Phase Space Cuts
- Couplings and Scales
- Standard-Model Parameters
- Total Cross Sections
- Resonance Decays
- Timelike Showers
- Spacelike Showers
- Multiple Interactions
- Beam Remnants
- Fragmentation
- Flavour Selection
- Particle Decays
- Bose-Einstein Effects
- Particle Data
- Error Checks
- Tunes

Study Output
- Four-Vectors
- Particle Properties
- Event Record

Event Information
- Event Statistics
- Histograms
- Event Analysis
- HepMC Interface

Link to Other Programs
- Les Houches Accord
- Access PYTHIA 6 Processes
- Semi-Internal Processes
- Semi-Internal Resonances
- Hadron-Level Standalone
- SUSY Les Houches Accord
- Parton Distributions
- External Decays
- User Hooks
- Random Numbers
- Implement New Showers

Reference Materiel
- Bibliography
- Glossary
- Version
Hard-process generation

Processes can be switched on with

\texttt{ProcessGroup:ProcessName = on}

or sometimes

\texttt{ProcessGroup:all = on}

<table>
<thead>
<tr>
<th>ProcessGroup</th>
<th>ProcessName</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoftQCD</td>
<td>minBias, elastic, singleDiffractive, doubleDiffractive</td>
</tr>
<tr>
<td>HardQCD</td>
<td>gg2gg, gg2qqbar, qq2qg, qq2qq, qqbar2gg, qqbar2qqbarNew, gg2ccbar, qqbar2ccbar, gg2bbbar, qqbar2bbbar</td>
</tr>
<tr>
<td>PromptPhoton</td>
<td>qg2qgamma, qqbar2ggamma, gg2ggamma, ffbar2gammagamma, gg2gammagamma</td>
</tr>
<tr>
<td>WeakBosonExchange</td>
<td>ff2ff(t:gmZ), ff2ff(t:W)</td>
</tr>
<tr>
<td>WeakSingleBoson</td>
<td>ffb2gmZ, ffb2W, ffb2ffbar(s:gm)</td>
</tr>
<tr>
<td>WeakDoubleBoson</td>
<td>ffb2gmZgmZ, ffb2ZW, ffb2WW</td>
</tr>
<tr>
<td>WeakBosonAndParton</td>
<td>qqbar2gmZg, qg2gmZq, ffb2gmZgm, fgm2gmZf, qqbar2Wg, qg2Wq, ffb2Wgm, fgm2Wf</td>
</tr>
<tr>
<td>Charmonium</td>
<td>gg2QQbar[3S1(1)]g, qg2QQbar[3PJ(8)]q, ...</td>
</tr>
<tr>
<td>Bottomonium</td>
<td>gg2QQbar[3S1(1)]g, gg2QQbar[3P2(1)]g, ...</td>
</tr>
<tr>
<td>ProcessGroup</td>
<td>ProcessName</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Top</td>
<td>gg2ttbar, qqbar2ttbar, qq2tq(t:W),</td>
</tr>
<tr>
<td></td>
<td>ffbare2ttbar(s:gmZ), fbar2ttqbar(s:W)</td>
</tr>
<tr>
<td>FourthBottom</td>
<td>gg2bPrimebPrimebar, qq2bPrimeq(t:W),</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>FourthTop</td>
<td>qqbar2tPrimetPrimebar, fbar2tPrimeqbar(s:W),</td>
</tr>
<tr>
<td>FourthPair</td>
<td>fbar2tPrimebPrimebar(s:W), fbar2tauPrimenuPrimebar(s:W)</td>
</tr>
<tr>
<td>HiggsSM</td>
<td>fbar2H, gg2H, fbar2HZ, ff2Hff(t:WW),</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>HiggsBSM</td>
<td>h, H and A as above, charged Higgs, pairs</td>
</tr>
<tr>
<td>SUSY</td>
<td>qqbar2chi0chi0 (SUSY barely begun)</td>
</tr>
<tr>
<td>NewGaugeBoson</td>
<td>fbar2gmZPrime, fbar2WPrime, fbar2R0</td>
</tr>
<tr>
<td>LeftRightSymmetry</td>
<td>fbar2ZR, fbar2WR, fbar2HLHL, ...</td>
</tr>
<tr>
<td>LeptoQuark</td>
<td>qL2LQ, q2LQ1, gg2LQLQbar, qqbar2LQLQbar</td>
</tr>
<tr>
<td>ExcitedFermion</td>
<td>dg2dStar, qq2uStarq, qqbar2Starmu, ...</td>
</tr>
<tr>
<td>ExtraDimensionsG*</td>
<td>gg2G*, qqbar2G*, ...</td>
</tr>
</tbody>
</table>

Can also use (and sometimes mix with)

- Les Houches Event Files
- Les Houches Accord-style runtime C++ interface
- Les Houches Accord runtime Fortran 77 interface
  (and that way runtime link to PYTHIA 6.4)
- semi-internal matrix elements and resonances
  (external matrix elements, internal phase space)
More on particle data

The static `ParticleDataTable` class contains info by PDG id code:

- `name(id)`, `hasAnti(id)`
- `spinType(id)`, `chargeType(id)`, `charge(id)`, `colType(id)`
- `m0(id)`, `mWidth(id)`, `mMin(id)`, `mMax(id)`, `tau0(id)`, ...

plus a vector of `DecayChannels` with

- `onMode()`, `bRatio()`, `meMode()`, `multiplicity()`, `product(i)`

User modifies by methods, `readString("...")` and `readFile("filename")` with commands `id:property = value` or `id:channel:property = value`.

Some special commands:

- `id:all` = `name` `antiName` `spinType` `chargeType` `colType` `m0` `mWidth` `mMin` `mMax` `tau0`
- `id:new` = `name` `antiName` `spinType` `chargeType` `colType` `m0` `mWidth` `mMin` `mMax` `tau0`
- `id:channel:all` = `onMode` `bRatio` `meMode` `products`
- `id:oneChannel` = `onMode` `bRatio` `meMode` `products`
- `id:addChannel` = `onMode` `bRatio` `meMode` `products`
- `id:onMode` = `onMode`
- `id:onIfAny` = `products` and `id:offIfAny` = `products`
- `id:onIfAll` = `products` and `id:offIfAll` = `products`
- `id:onIfMatch` = `products` and `id:offIfMatch` = `products`
Useful commands:
- `pythia.particleData.listAll()` : complete list
- `pythia.particleData.listChanged()` : only changed ones
- `pythia.particleData.list(id)`: only one (or `vector<int>`)  

-------- PYTHIA Particle Data Table (changed only) -------------------------

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>antiName</th>
<th>spn</th>
<th>chg</th>
<th>col</th>
<th>m0</th>
<th>mWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>pi0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.13498</td>
<td>0.00000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.00000</td>
<td>2.51000e-05</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>0.9879900</td>
<td>0</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0.0119800</td>
<td>11</td>
<td>22</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1</td>
<td>0.0000300</td>
<td>13</td>
<td>11</td>
<td>-11</td>
</tr>
<tr>
<td>223</td>
<td>omega</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.78259</td>
<td>0.00849</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>0.8924000</td>
<td>1</td>
<td>211</td>
<td>-211</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0.0892800</td>
<td>0</td>
<td>22</td>
<td>111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1</td>
<td>0.0170000</td>
<td>3</td>
<td>211</td>
<td>-211</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>1</td>
<td>0.0004900</td>
<td>0</td>
<td>221</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>1</td>
<td>0.0000700</td>
<td>0</td>
<td>111</td>
<td>111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>1</td>
<td>0.0005900</td>
<td>0</td>
<td>111</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>1</td>
<td>0.0001000</td>
<td>0</td>
<td>111</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>1</td>
<td>0.0000700</td>
<td>0</td>
<td>11</td>
<td>-11</td>
</tr>
</tbody>
</table>

-------- End PYTHIA Particle Data Table ------------------------------------
The Particle class in the event record

Each Particle object stores the properties:

- **id()**: particle identity, by PDG codes.
- **status()**: status code. Provides info on where and why a given particle was produced. Negative code = no longer existing particle.
- **mother1(), mother2()**: first and last mother indices.
- **daughter1(), daughter2()**: first and last daughter indices.
- **col(), acol()**: colour and anticolour tags, Les Houches Accord.
- **px(), py(), pz(), e()**: four-momentum components (in GeV).
- **m()**: mass.
- **scale()**: scale at which a parton was produced; model-specific.
- **xProd(), yProd(), zProd(), tProd()**: production vertex (in mm).
- **tau()**: proper lifetime.

Methods above can also be used, with argument, for setting properties. Many further methods for extraction only, e.g. for rapidity. Also pointer to ParticleDataTable entry; gives e.g. name() and charge().
The Event class

Two Event objects inside a Pythia object:
- **process**: hard subprocess, roughly like Les Houches.
- **event**: complete event history.

An Event $\approx$ a vector<Particle>
e.g. pythia.event[i].id() = identity of i’th particle

index 0 = event-as-a-whole; not really part of history
- $\Rightarrow$ throw line 0 for HepMC conversion
- $\Rightarrow$ mother/daughter = 0 $\Rightarrow$ empty

Specific methods include:
- **size()**: $0 \leq i <$ event.size().
- **list()**: provide event listing.
- **motherList(i), daughterList(i), sisterList()**: a vector<int> of mothers, daughters, sisters.
- **iTopCopy(i), iBotCopy(i)**: top or bottom “carbon copy”.

But no methods to edit the event.

Further: info on junctions, subsystems (multiple interactions), …
Sample event listings

First with `pythia.process.list()`, truncated to fit:

```
-------- PYTHIA Event Listing (hard process) ---------------------------------------

<table>
<thead>
<tr>
<th>no</th>
<th>id</th>
<th>name</th>
<th>status</th>
<th>mothers</th>
<th>daughters</th>
<th>colours</th>
<th>p_x</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>90</td>
<td>(system)</td>
<td>-11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>2212</td>
<td>(p+)</td>
<td>-12</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2212</td>
<td>(p+)</td>
<td>-12</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>-2</td>
<td>(ubar)</td>
<td>-21</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>(u)</td>
<td>-21</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>-6</td>
<td>(tbar)</td>
<td>-22</td>
<td>3</td>
<td>0</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>(t)</td>
<td>-22</td>
<td>3</td>
<td>0</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>-24</td>
<td>(W-)</td>
<td>-22</td>
<td>5</td>
<td>0</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>-5</td>
<td>bbar</td>
<td>23</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>24</td>
<td>(W+)</td>
<td>-22</td>
<td>6</td>
<td>0</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>b</td>
<td>23</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>102</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>s</td>
<td>23</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>-4</td>
<td>cbar</td>
<td>23</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>-11</td>
<td>e+</td>
<td>23</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>12</td>
<td>nu_e</td>
<td>23</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Charge sum: 0.000  Momentum sum: 0.000

-------- End PYTHIA Event Listing -----------------------------------------------
```

next with `pythia.event.list()`, omissions to fit:
## PYTHIA Event Listing

### (complete event)

<table>
<thead>
<tr>
<th>no</th>
<th>id</th>
<th>name</th>
<th>status</th>
<th>mothers</th>
<th>daughters</th>
<th>colours</th>
<th>p_x</th>
<th>p_y</th>
<th>p_z</th>
<th>e</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>90</td>
<td>(system)</td>
<td>-11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>1</td>
<td>2212</td>
<td>(p+)</td>
<td>-12</td>
<td>0</td>
<td>0</td>
<td>279</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.000</td>
<td>7000.000</td>
</tr>
<tr>
<td>2</td>
<td>2212</td>
<td>(p+)</td>
<td>-12</td>
<td>0</td>
<td>0</td>
<td>280</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.000</td>
<td>-7000.000</td>
</tr>
<tr>
<td>3</td>
<td>-2</td>
<td>(ubar)</td>
<td>-21</td>
<td>8</td>
<td>0</td>
<td>5</td>
<td>6</td>
<td>102</td>
<td>0</td>
<td>0.000</td>
<td>1042.471</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>(u)</td>
<td>-21</td>
<td>8</td>
<td>0</td>
<td>5</td>
<td>6</td>
<td>102</td>
<td>0</td>
<td>0.000</td>
<td>54.594</td>
</tr>
<tr>
<td>5</td>
<td>-6</td>
<td>(tbar)</td>
<td>-22</td>
<td>3</td>
<td>4</td>
<td>9</td>
<td>9</td>
<td>101</td>
<td>-73.897</td>
<td>-53.244</td>
<td>-174.768</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>(t)</td>
<td>-22</td>
<td>3</td>
<td>4</td>
<td>10</td>
<td>10</td>
<td>102</td>
<td>73.897</td>
<td>53.244</td>
<td>813.108</td>
</tr>
<tr>
<td>7</td>
<td>-2</td>
<td>(ubar)</td>
<td>-42</td>
<td>12</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>101</td>
<td>0.000</td>
<td>54.594</td>
<td>54.594</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>(u)</td>
<td>-41</td>
<td>13</td>
<td>13</td>
<td>11</td>
<td>4</td>
<td>104</td>
<td>0.000</td>
<td>-1191.549</td>
<td>1191.549</td>
</tr>
<tr>
<td>9</td>
<td>-6</td>
<td>(tbar)</td>
<td>-44</td>
<td>5</td>
<td>5</td>
<td>14</td>
<td>14</td>
<td>101</td>
<td>-71.565</td>
<td>-51.768</td>
<td>-210.234</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>(t)</td>
<td>-44</td>
<td>6</td>
<td>6</td>
<td>15</td>
<td>15</td>
<td>102</td>
<td>82.715</td>
<td>58.828</td>
<td>926.573</td>
</tr>
<tr>
<td>11</td>
<td>21</td>
<td>(g)</td>
<td>-43</td>
<td>8</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>104</td>
<td>-11.150</td>
<td>-7.060</td>
<td>-13.198</td>
</tr>
<tr>
<td>25</td>
<td>21</td>
<td>(g)</td>
<td>-51</td>
<td>23</td>
<td>0</td>
<td>37</td>
<td>37</td>
<td>106</td>
<td>19.037</td>
<td>28.329</td>
<td>38.331</td>
</tr>
<tr>
<td>26</td>
<td>21</td>
<td>(g)</td>
<td>-51</td>
<td>23</td>
<td>0</td>
<td>39</td>
<td>39</td>
<td>101</td>
<td>6.832</td>
<td>-19.532</td>
<td>2.861</td>
</tr>
<tr>
<td>27</td>
<td>-6</td>
<td>(tbar)</td>
<td>-52</td>
<td>20</td>
<td>20</td>
<td>34</td>
<td>34</td>
<td>101</td>
<td>-88.187</td>
<td>-52.597</td>
<td>-231.302</td>
</tr>
<tr>
<td>44</td>
<td>21</td>
<td>(g)</td>
<td>-31</td>
<td>48</td>
<td>0</td>
<td>46</td>
<td>47</td>
<td>114</td>
<td>113</td>
<td>0.000</td>
<td>23.294</td>
</tr>
<tr>
<td>45</td>
<td>1</td>
<td>(d)</td>
<td>-31</td>
<td>49</td>
<td>49</td>
<td>46</td>
<td>47</td>
<td>113</td>
<td>0.000</td>
<td>-255.118</td>
<td>255.118</td>
</tr>
<tr>
<td>46</td>
<td>21</td>
<td>(g)</td>
<td>-33</td>
<td>44</td>
<td>45</td>
<td>50</td>
<td>50</td>
<td>114</td>
<td>115</td>
<td>2.524</td>
<td>5.061</td>
</tr>
<tr>
<td>47</td>
<td>1</td>
<td>(d)</td>
<td>-33</td>
<td>44</td>
<td>45</td>
<td>51</td>
<td>51</td>
<td>115</td>
<td>0.000</td>
<td>-2.524</td>
<td>-5.061</td>
</tr>
<tr>
<td>378</td>
<td>2</td>
<td>(u)</td>
<td>-63</td>
<td>1</td>
<td>0</td>
<td>492</td>
<td>492</td>
<td>113</td>
<td>0.000</td>
<td>-3.191</td>
<td>-0.512</td>
</tr>
<tr>
<td>379</td>
<td>2101</td>
<td>(ud_0)</td>
<td>-63</td>
<td>1</td>
<td>0</td>
<td>492</td>
<td>492</td>
<td>0</td>
<td>113</td>
<td>-0.427</td>
<td>-1.024</td>
</tr>
<tr>
<td>380</td>
<td>2</td>
<td>(u)</td>
<td>-63</td>
<td>1</td>
<td>0</td>
<td>493</td>
<td>493</td>
<td>108</td>
<td>0.000</td>
<td>-1.118</td>
<td>56.936</td>
</tr>
<tr>
<td>381</td>
<td>-3</td>
<td>(sbar)</td>
<td>-63</td>
<td>1</td>
<td>0</td>
<td>519</td>
<td>519</td>
<td>0</td>
<td>117</td>
<td>-0.382</td>
<td>1364.384</td>
</tr>
<tr>
<td>486</td>
<td>-11</td>
<td>e+</td>
<td>23</td>
<td>441</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7.949</td>
<td>-14.875</td>
</tr>
<tr>
<td>487</td>
<td>12</td>
<td>nu_e</td>
<td>23</td>
<td>441</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>70.533</td>
<td>75.395</td>
</tr>
<tr>
<td>502</td>
<td>1</td>
<td>(d)</td>
<td>-71</td>
<td>342</td>
<td>342</td>
<td>505</td>
<td>508</td>
<td>115</td>
<td>0</td>
<td>-3.404</td>
<td>-4.046</td>
</tr>
<tr>
<td>503</td>
<td>21</td>
<td>(g)</td>
<td>-71</td>
<td>367</td>
<td>367</td>
<td>505</td>
<td>508</td>
<td>181</td>
<td>115</td>
<td>-0.384</td>
<td>-0.368</td>
</tr>
<tr>
<td>504</td>
<td>-2</td>
<td>(ubar)</td>
<td>-71</td>
<td>370</td>
<td>370</td>
<td>505</td>
<td>508</td>
<td>0</td>
<td>181</td>
<td>-3.167</td>
<td>-0.517</td>
</tr>
<tr>
<td>505</td>
<td>311</td>
<td>(K0)</td>
<td>-83</td>
<td>502</td>
<td>504</td>
<td>789</td>
<td>789</td>
<td>0</td>
<td>0</td>
<td>-2.046</td>
<td>-0.406</td>
</tr>
<tr>
<td>506</td>
<td>331</td>
<td>(eta')</td>
<td>-83</td>
<td>502</td>
<td>504</td>
<td>941</td>
<td>942</td>
<td>0</td>
<td>0</td>
<td>-1.070</td>
<td>-2.000</td>
</tr>
<tr>
<td>507</td>
<td>-323</td>
<td>(K+*)</td>
<td>-83</td>
<td>502</td>
<td>504</td>
<td>790</td>
<td>791</td>
<td>0</td>
<td>0</td>
<td>-2.736</td>
<td>-2.575</td>
</tr>
<tr>
<td>508</td>
<td>111</td>
<td>(pi0)</td>
<td>-84</td>
<td>502</td>
<td>504</td>
<td>943</td>
<td>944</td>
<td>0</td>
<td>0</td>
<td>-1.102</td>
<td>0.050</td>
</tr>
<tr>
<td>789</td>
<td>130</td>
<td>K_L0</td>
<td>91</td>
<td>505</td>
<td>505</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-2.046</td>
<td>-0.406</td>
</tr>
<tr>
<td>790</td>
<td>-311</td>
<td>(Kbar0)</td>
<td>-91</td>
<td>507</td>
<td>0</td>
<td>932</td>
<td>932</td>
<td>0</td>
<td>0</td>
<td>-0.900</td>
<td>-1.003</td>
</tr>
<tr>
<td>791</td>
<td>-211</td>
<td>pi-</td>
<td>91</td>
<td>507</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1.836</td>
<td>-1.571</td>
</tr>
<tr>
<td>792</td>
<td>-211</td>
<td>pi-</td>
<td>91</td>
<td>516</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.117</td>
<td>-0.161</td>
</tr>
<tr>
<td>793</td>
<td>111</td>
<td>(pi0)</td>
<td>-91</td>
<td>516</td>
<td>0</td>
<td>1069</td>
<td>1070</td>
<td>0</td>
<td>0</td>
<td>-0.431</td>
<td>-0.098</td>
</tr>
<tr>
<td>794</td>
<td>2212</td>
<td>p+</td>
<td>91</td>
<td>537</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1.175</td>
<td>0.093</td>
</tr>
<tr>
<td>795</td>
<td>211</td>
<td>pi+</td>
<td>91</td>
<td>537</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.414</td>
<td>0.352</td>
</tr>
<tr>
<td>1316</td>
<td>22</td>
<td>gamma</td>
<td>91</td>
<td>1313</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1.574</td>
<td>0.014</td>
</tr>
<tr>
<td>1317</td>
<td>22</td>
<td>gamma</td>
<td>91</td>
<td>1313</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-0.887</td>
<td>0.068</td>
</tr>
</tbody>
</table>

**Charge sum:** 2.000  
**Momentum sum:** -0.000  
---

**End PYTHIA Event Listing**
You can use `pythia.info.method()` to extract one-of-a-kind information, such as:

- **idA(), idB(), eCM()**: incoming beams and cm energy.
- **name(), code()**: the name and code of the subprocess.
- **id1(), id2(), x1(), x2()**: the identities and $x$ fractions of the two partons coming in to the hard subprocess.
- **pdf1(), pdf2(), Q2Fac()**: parton densities $x f(x, Q^2)$ evaluated for the two incoming partons, and the associated $Q^2$ scale.
- **alphaS(), alphaEM(), Q2Ren()**: $\alpha_s$, $\alpha_{em}$ and their $Q^2$ scale.
- **mHat(), sHat(), tHat(), uHat()**: the invariant mass of the hard subprocess and the Mandelstam variables.
- **pTHat(), thetaHat(), phiHat()**: transverse momentum and polar and azimuthal scattering angles of the hard subprocess.
- **bMI(), nMI()**: impact parameter (rescaled) and number of multiple interactions.
- **list()**: list some information on output.
- **sigmaGen(), sigmaErr()**: the process-summed estimated cross section and its estimated statistical error, in mb.
Event analysis

Four-vectors in a class `Vec4`, with overloaded operators.

A small package for one-dimensional histograms:
- Book with `Hist name( title, numberOfBins, xMin, xMax);`
  or `Hist name; name.book(title, numberOfBins, xMin, xMax);`
- Fill with `name.fill( xValue, weight);` with default `weight = 1`
- Print with `cout << name;`
- Overloaded operators for addition, multiplication, ...

Sphericity analysis (similarly thrust):
- Instantiate with `Sphericity sph( power, select);`
- Analyze with `sph.analyze( event);`
- Info with `sph.sph(), sph.EigenVector(i), sph.list(), ...`

Cone jet finder a la UA1 (PYCELL) (similarly Lund/JADE/Durham):
- Instantiate with `CellJet cellJet( etaMax, nEta, nPhi,
  select, smear, resolution, upperCut, threshold);`
- Analyze with `cellJet.analyze(event, eTjetMin, coneRadius, eTseed);`
- Info with `cellJet.size(), cellJet.eT(i), cellJet.list(), ...`
Statistics

Output from `pythia.statistics()` (some blanks removed for space):

*-------- PYTHIA Event and Cross Section Statistics ---------------------------------------*

<table>
<thead>
<tr>
<th>Subprocess Code</th>
<th>Number of events</th>
<th>sigma ± delta (estimated) (mb)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tried</td>
<td>Selected</td>
</tr>
<tr>
<td>g g -&gt; g g</td>
<td>111</td>
<td>502</td>
</tr>
<tr>
<td>g g -&gt; q qbar (uds)</td>
<td>112</td>
<td>2</td>
</tr>
<tr>
<td>q g -&gt; q g</td>
<td>113</td>
<td>247</td>
</tr>
<tr>
<td>q q(bar)' -&gt; q q(bar)'</td>
<td>114</td>
<td>24</td>
</tr>
<tr>
<td>q qbar -&gt; g g</td>
<td>115</td>
<td>1</td>
</tr>
<tr>
<td>q qbar -&gt; q’ qbar’ (uds)</td>
<td>116</td>
<td>0</td>
</tr>
<tr>
<td>g g -&gt; c cbar</td>
<td>121</td>
<td>1</td>
</tr>
<tr>
<td>g g -&gt; b bbar</td>
<td>123</td>
<td>2</td>
</tr>
<tr>
<td>sum</td>
<td>779</td>
<td>100</td>
</tr>
</tbody>
</table>

*-------- End PYTHIA Event and Cross Section Statistics ---------------------------------------*

*-------- PYTHIA Error and Warning Messages Statistics ---------------------------------------*

<table>
<thead>
<tr>
<th>times</th>
<th>message</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Error in Pythia::next: hadronLevel failed; try again</td>
</tr>
<tr>
<td>3</td>
<td>Error in StringFragmentation::fragmentToJunction: caught in junction flavour loop</td>
</tr>
<tr>
<td>3</td>
<td>Warning in ParticleDataEntry::initBWmass: switching off width</td>
</tr>
</tbody>
</table>

*-------- End PYTHIA Error and Warning Messages Statistics ---------------------------------------*
Link to other program

PYTHIA is standalone, but several possibilities to link to it.

Possibilities similar to PYTHIA 6.4:
- Input from Les Houches Accord & Les Houches Event Files
- Output to HepMC event format (more robust than PYTHIA 6!?)
- SUSY Les Houches Accord (input file with masses, couplings, \ldots)
- Link to external decays, e.g. for $\tau$ and $B$.
- Link to LHAPDF version 5.3.0 or later, or to your own PDF.

New possibilities, based on derived classes and pointers to them:
- Semi-internal process: write derived matrix-element class,
  \begin{verbatim}
  SigmaProcess* mySigma = new MySigma();
  pythia.setSigmaPtr( mySigma);
  \end{verbatim}
  and let PYTHIA do phase space integration, process mixing, \ldots
- Semi-internal resonance in same style: calculate partial widths
- Link to external random-number generator.
- Link to external shower, e.g. VINCIA for FSR.
- User hooks: veto events early on or reweight cross section.
Sample Main Programs

- `main01.cc`: charged multiplicity distribution
- `main02.cc`: $Z^0 p_\perp$ spectrum
- `main03.cc` & `main03.cmnd`: single-particle analysis in jet events
- `main04.cc` & `main04.cmnd`: tests of event properties
- `main05.cc`: cone-jet analysis of LHC events
- `main06.cc` & `main06.cmnd`: study elastic/diffractive events
- `main07.cc` & `main07.cmnd`: study minimum-bias events
- `main08.cc` & `main08.cmnd`: combine results of subruns in $p_\perp$ bins
- `main09.cc`: LEP events with sphericity/thrust/jetfinder analysis
- `main10.cc`: use UserHooks to interact with generation process
- `main11.cc`: set two hard interactions in the same event
- `main12.cc` & `ttbar.lhe`: input from a Les Houches Event File
- `main13.cc` & `ttbar.lhe` & `ttbar2.lhe`: input from two Les Houches Event Files; mix with internal processes
- `main14.cc`: compare several cross sections with PYTHIA 6.4 values
- `main15.cc`: redo B decays several times for each event
- main16.cc: user analysis class; command-line input file
- main17.cc: Pythia wrapper class; command-line input file
- main21.cc: input of parton configurations for hadronization only
- main22.cc & main22.cmnd & main22.spc: SUSY with SLHA input
- main23.cc: link an external decay handler
- main24.cc: link an external random number generator
- main25.cc: link an external process for internal use
- main26.cc: link an external resonance and process for internal use
- main31.cc & main31.cmnd: simple output to HepMC event file
- main32.cc & main32.cmnd: streamlined production to HepMC; command-line input and output files
- main41.cc: test shapes of PDF’s in LHAPDF
- main42.cc: compare event properties for different LHAPDF PDF’s
- main51.cc: runtime LHA link to PYTHIA 6.4
- main52.cc & main52.ccmnd & main52.fcmnd: ditto with input files
- main53.f: (Fortran!) have PYTHIA 6.4 generate an LHEF
- main54.cc & main54.cmnd: input from PYTHIA 6.4 and output to HepMC
License and Acknowledgements

Based on MCnet discussions during the spring there is a HERWIG++/SHERPA/PYTHIA/THEPEG/ARIADNE/… agreement:

- Our programs are licensed under the GPL version 2.
- Please respect the MCnet Guidelines for Event Generator Authors and Users.
  1. The integrity of the program should be respected.
     - report bugs & fixes to authors — don’t create own forks
     - redistribute a program in its entirety, not piecemeal
  2. The program and its physics should be properly cited when used for academic publications.
     - cite manuals, but also physics articles of special relevance
     - cite all programs used, commensurate with importance for study
     - document version/parameters for reproducibility of publications

Makefiles, configure scripts & HepMC interface by Mikhail Kirsanov.
Conversion to PHP files by Ben Lloyd.
Win32/NMAKE by Bertrand Bellenot.
Extended Higgs sector by Marc Montull.
Some c/b decay tables from LHCb & DELPHI.
Outlook

We are now in a chicken-and-egg situation: the user community needs a mature program; but PYTHIA 8 will only mature if there is an active user community.

So please . . .
- implement in your experimental frameworks
- find volunteers to act as guinea pigs
- do some small-scale “production runs”
- report back problems & wishes (within reason)

Don’t throw away PYTHIA 6.4 just yet!
- 8.1 still can’t do everything 6.4 can
- 8.1 still needs testing and tuning

As new features are introduced, 8.1 will become the obvious choice:
- improved multiple interactions
- more matrix-element matching
- ???