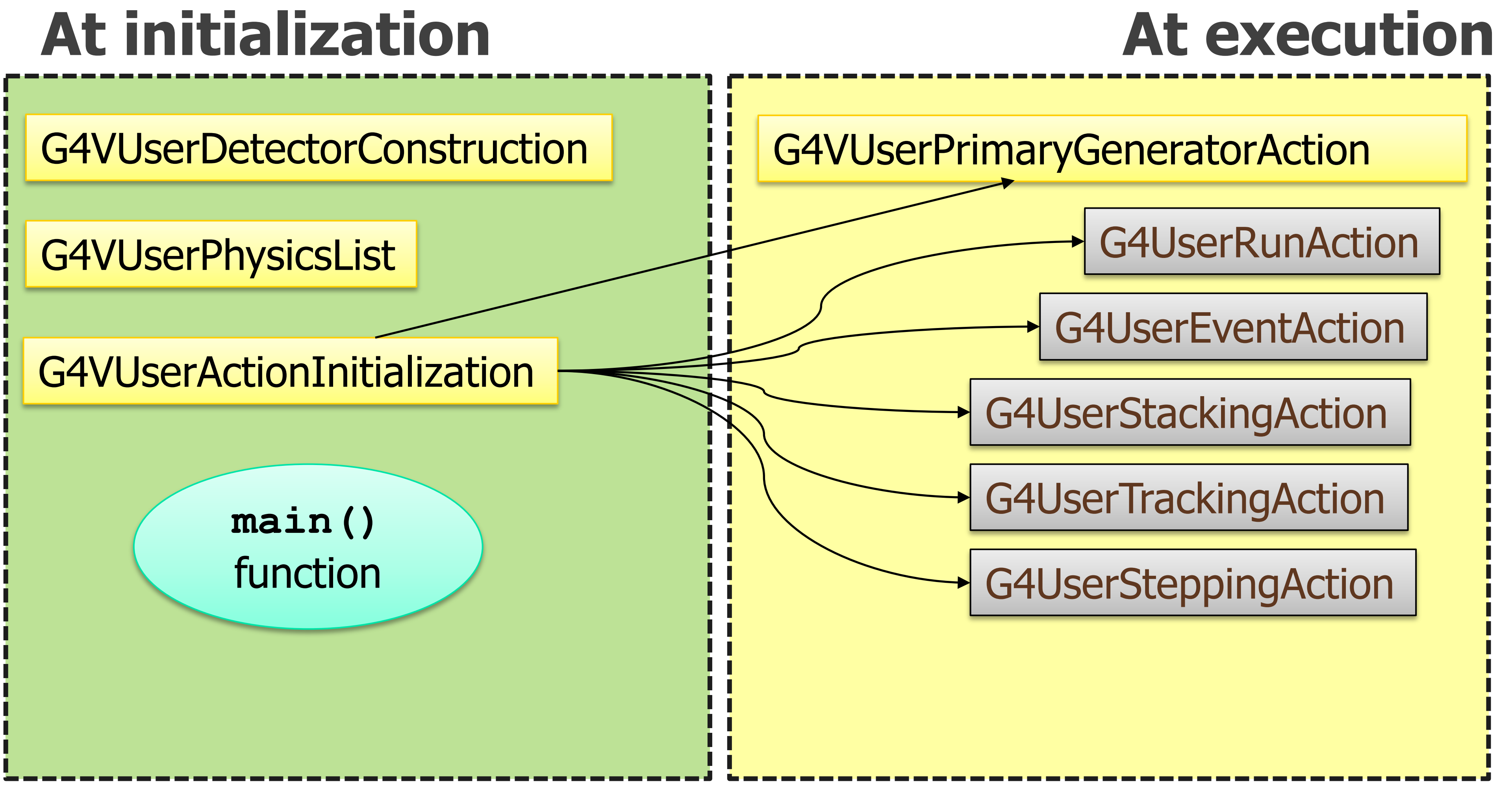
Geant4 physics

Luciano Pandola
INFN – Laboratori Nazionali del Sud

A lot of material by G.A.P. Cirrone and J. Pipek

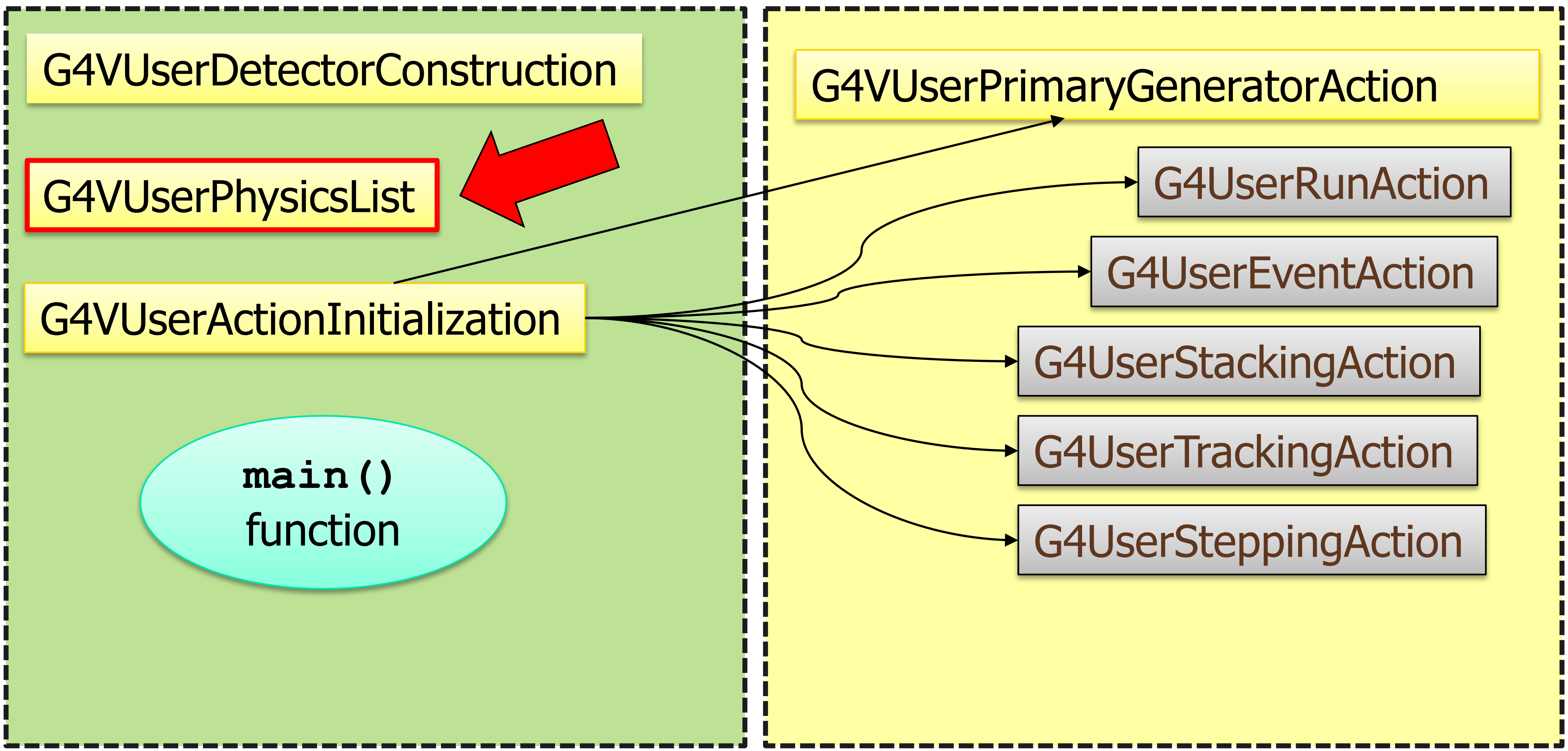
Geant4 Course, XX Seminar on Software for Nuclear, Subnuclear and Applied Physics, Alghero, June 5th- 9th, 2023

Mandatory (and optional) user classes



Mandatory (and optional) user classes

At initialization **At execution**



Outlook

- Physics in Geant4 – motivation
- Particles & processes
- Physics lists
- Production cuts
- Electromagnetic/hadronic physics

"Shouldn't there be just one universal and complete physics description?"

No.

Physics – the challenge

- Huge amount of different processes for various purposes (*only a handful relevant*)
- Competing descriptions of the same physics phenomena *(necessary to choose)*
  - fundamentally different **approaches**
  - balance between **speed** and **precision**
  - different **parameterizations**
- Hypothetical processes & exotic physics

Solution: Atomistic approach with modular **physics lists**

Part I: Particles and Processes

Particles: basic concepts

- There are three levels of class to describe particles in Geant4:
- **`G4ParticleDefinition`**
  - Particle static properties: name, mass, spin, PDG number, etc.
- **`G4DynamicParticle`**
  - Particle dynamic state: energy, momentum, polarization, etc.
- **`G4Track`**
  - Information for tracking in a detector simulation: position, step, current volume, track ID, parent ID, etc.

Particles in Geant4

- Particle Data Group (PDG) particles
- Optical photons (different from gammas!)
- Special particles: geantino and charged geantino
  - Only transported in the geometry (no interactions)
  - Charged geantino also feels the EM fields
- Short-lived particles ($\tau < 10^{-14}$ s) are not transported by Geant4 *(decay applied)*
- Light ions (as deuterons, tritons, alphas)
- Heavier ions represented by a single class: `G4Ions`

Particle name	Class name	Name (in GPS...)	PDG
electron	G4Electron	e-	11
positron	G4Positron	e+	-11
muon +/-	G4MuonPlus G4MuonMinus	mu+ mu-	-13 13
tauon +/-	G4TauPlus G4TauMinus	tau+ tau-	-15 15
electron (anti)neutrino	G4NeutrinoE G4AntiNeutrinoE	nu_e anti_nu_e	12 -12
muon (anti)neutrino	G4NeutrinoMu G4AntiNeutrinoMu	nu_mu anti_nu_mu	14 -14
tau (anti)neutrino	G4NeutrinoTau G4AntiNeutrinoTau	nu_tau anti_nu_tau	16 -16
photon (γ, X)	G4Gamma	gamma	22
photon (optical)	G4OpticalPhoton	opticalphoton	(0)
geantino	G4Geantino	geantino	(0)
charged geantino	G4ChargedGeantino	chargedgeantino	(0)

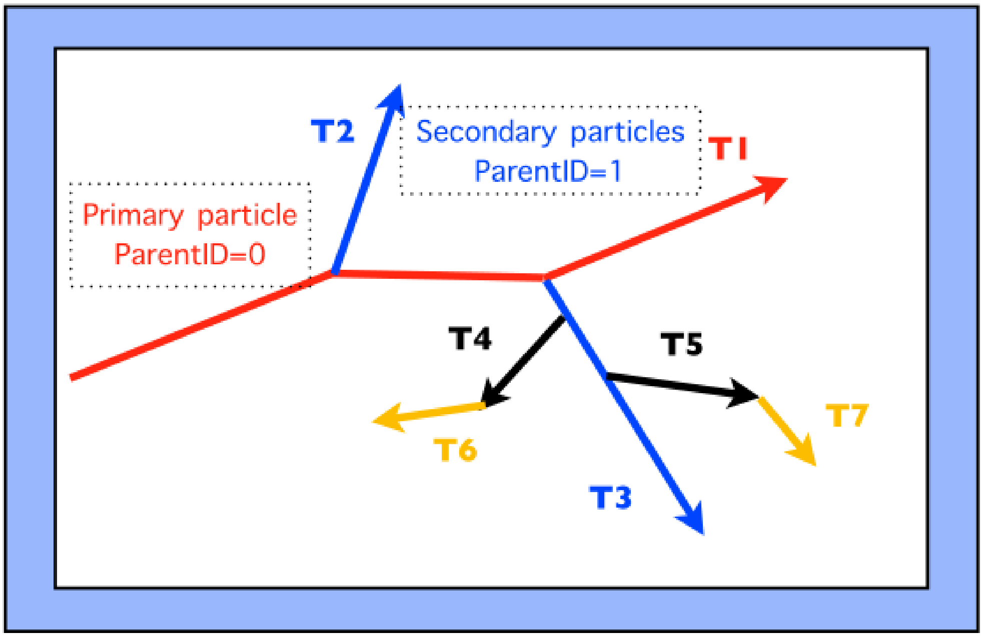
Processes

How do particles interact with materials?

- Responsibilities:
  - decide when and where an **interaction** occurs
    - GetPhysicalInteractionLength...() → limit the step
    - this requires a cross section
    - for the transportation process, the distance to the nearest object
  - generate the **final state** of the interaction
    - changes momentum, generates secondaries, etc.
    - method: DoIt...()
    - this requires a model of the physics

Part II: Tracking and cuts

Geant4 way of tracking



- **Force step** at geometry boundaries
- All AlongStep processes **co-work**, the PostStep **compete** (= only one selected)
- Call AtRest actions for particles at rest

- Secondaries saved at the top of the stack: tracking order follows **'last in first out'** rule:
  T1 → T3 → T5 → T7 → T4 → T6 → T2

Tracking verbosity

UI command: **`/tracking/verbose 1`**

```
***********************************************************************************************
* G4Track Information:   Particle = gamma,   Track ID = 1,   Parent ID = 0
***********************************************************************************************

Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0     47.4      -53     -150         6        0        0         0    Envelope initStep
    1     47.4      -53      -58     0.844        0       92        92    Envelope compt
    2      -46     15.9     5.55      0.47        0      132       224    Envelope compt
    3     -100     6.37    -3.62      0.47        0     55.6       280       World
Transportation
    4     -120     2.84    -7.02      0.47        0     20.6       301  OutOfWorld
Transportation
***********************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 3,   Parent ID = 1
***********************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0      -46     15.9     5.55     0.375        0        0         0    Envelope initStep
    1    -46.1     16.4     5.98    0.0482    0.327     1.16      1.16    Envelope eIoni
    2    -46.1     16.3     5.98         0   0.0482   0.0408       1.2    Envelope eIoni
```

Primary γ

Compton e$^-$

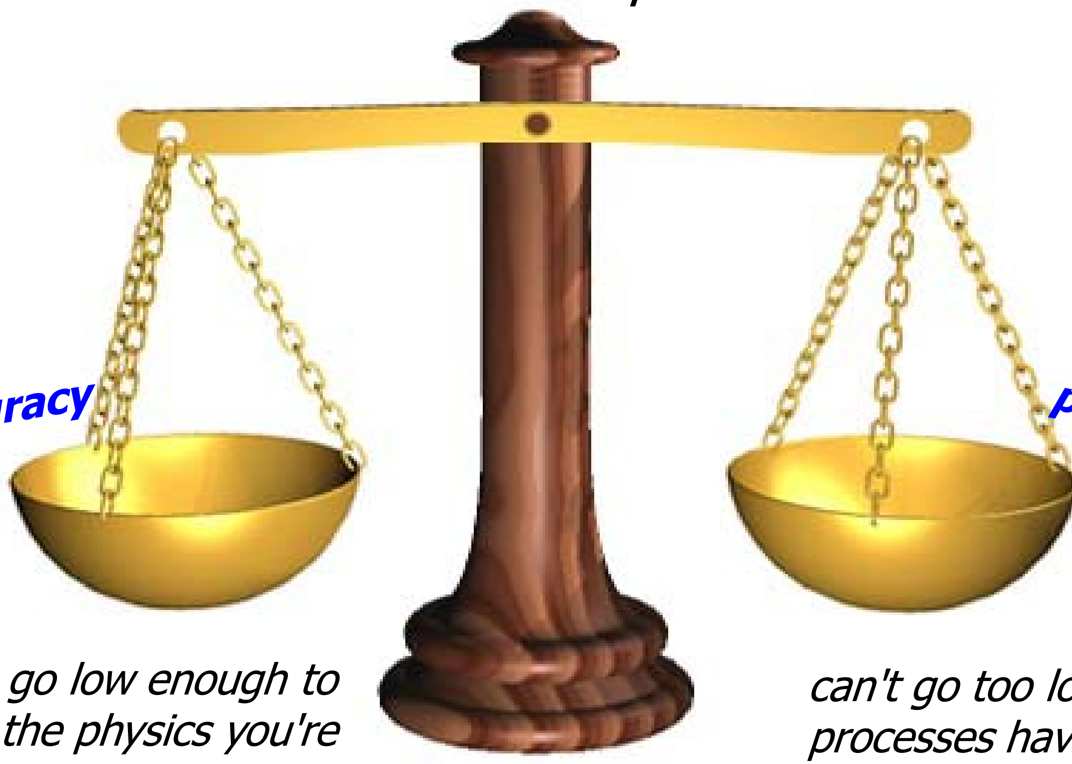
Geant4 production cuts

- Geant4 does not have tracking cuts
  - All tracks are followed down to zero energy
    - ..or until they leave the world volume or are destroyed in interactions
  - Could be implemented manually by the user
- Geant4 uses only a **production cut → "range production threshold"**
  - i.e. cuts deciding whether a **secondary** particle to be produced or not
    - AlongStep vs. PostStep
    - Applies **only** to: γ from bremsstrahlung, e^- from ionization and low-energy protons from hadronic elastic scattering
  - This threshold is a distance, not an energy
    - Particles unable to travel at least the range cut value are not produced
- One production threshold is uniformly set
  - Sets the **"spatial accuracy"** of the simulation
- Production threshold is internally converted to the energy threshold, depending on *particle* type and *material*

Production cut

- Key ingredient of the mixed MC: threshold

the best compromise



accuracy

performance

need to go low enough to get the physics you're interested in

can't go too low because some processes have infrared divergence causing huge CPU time

Cuts – UI commands

```
# Universal cut (whole world, all particles)
/run/setCut 10 mm

# Override low-energy limit
/cuts/setLowEdge 100 eV

# Set cut for a specific particle (whole world)
/run/setCutForAGivenParticle gamma 0.1 mm

# Set cut for a region (all particles)
/run/setCutForARegion myRegion 0.01 mm

# Print a summary of particles/regions/cuts
/run/dumpCouples
```

Part III: Physics lists & Co.

A physics list: what it is, what it does

- One instance per application
  - registered to run manager in `main()`
  - inheriting from `G4VUserPhysicsList`
- Responsibilities
  - all **particle types** (electron, proton, gamma, ...)
  - all **processes** (photoeffect, bremsstrahlung, ...)
  - all **process parameters** (...)
  - **production cuts** (e.g. 1 mm for electrons, ...)

G4VUserPhysicsList

- All physics lists **must** derive from this class
  - And then be registered to the G4(MT)RunManager
  - Mandatory class in Geant4

```
class MyPhysicsList: public G4VUserPhysicsList {
public:
MyPhysicsList();
~MyPhysicsList();
void ConstructParticle();
void ConstructProcess();
void SetCuts();
}
```

- User must implement the following (purely virtual) methods:
  - `ConstructParticle()`, `ConstructProcess()`
- Optional Virtual method:
  - `SetCuts()` (used to be purely virtual up to 10.2)

Three ways to get a physics list

- **Manual**: Write your own class, to specify all particles & processes that may occur in the simulation (very flexible, but difficult)
- **Physics constructors**: Combine your physics from pre-defined sets of particles and processes. Still you define your own class – modular physics list (easier)
- **Reference physics lists**: Take one of the pre-defined physics lists. You don't create any class (easy)

Way 1

Derived class from `G4VUserPhysicsList`

- Implement 3 methods:

```
class MyPhysicsList : public G4VUserPhysicsList {
public:
    // ...
    void ConstructParticle();  // pure virtual
    void ConstructProcess();   // pure virtual
    void SetCuts();
    // ...
}
```

Advantage: most flexible

Disadvantages:

- most verbose
- most difficult to get right

G4VUserPhysicsList: implementation

- **ConstructParticle()**
  - choose the particles you need in your simulation, define **all of them** here
- **ConstructProcess()**
  - for each particle, assign all the physics processes relevant to your simulation
- **SetCuts()**
  - set the range cuts for secondary production for processes with infrared divergence

Way 2

G4VModularPhysicsList

- Similar structure as **G4VUserPhysicsList** (same methods to override – though not necessary):

```
class MyPhysicsList : public G4VModularPhysicsList {
public:
    MyPhysicsList();              // define physics constructors
    void ConstructParticle();     // optional
    void ConstructProcess();      // optional
    void SetCuts();               // optional
}
```

Differences to "manual" way:

- Particles and processes typically handled by **physics constructors** (still customizable)
- **Transportation** automatically included

Physics constructors (1)

- "**Building blocks**" of a modular physics list
- Inherit from **`G4VPhysicsConstructor`**
- Defines **`ConstructParticle`**`()` and **`ConstructProcess()`**
  - to be fully imported in modular list (behaving in the same way)
- **`GetPhysicsType()`**
  - enables switching physics of the same type, if possible (see next slide)

Physics constructors (2)

- Huge set of pre-defined ones
  - **EM**: Standard, Livermore, Penelope
  - **Hadronic inelastic:** QGSP_BIC, FTFP_Bert, ...
  - **Hadronic elastic:** G4HadronElasticPhysics, ...
  - ... (decay, optical physics, EM extras, ...)
- You can implement your own *(of course)* by inheriting from the **`G4VPhysicsConstructor`** class

`Code: $G4INSTALL/source/physics_lists/constructors`

How to use physics constructors

Add **physics constructor** in the class **constructor**:

```
MyModularList::MyModularList() {
    // Hadronic physics
    RegisterPhysics(new G4HadronElasticPhysics());
    RegisterPhysics(new G4HadronPhysicsFTFP_BERT_TRV());
    // EM physics
    RegisterPhysics(new G4EmStandardPhysics());
}
```

This already works and no further method overriding is necessary ☺

Way 3

Reference physics lists

- Pre-defined ("plug-and-play") physics lists
  - already containing a **complete set** of particles & processes (that work together)
  - targeted at specific area of interest (HEP, medical physics, ...)
  - constructed as modular physics lists, built on top of **physics constructors**
  - customizable (by calling appropriate methods before initialization)

Using a reference physics list

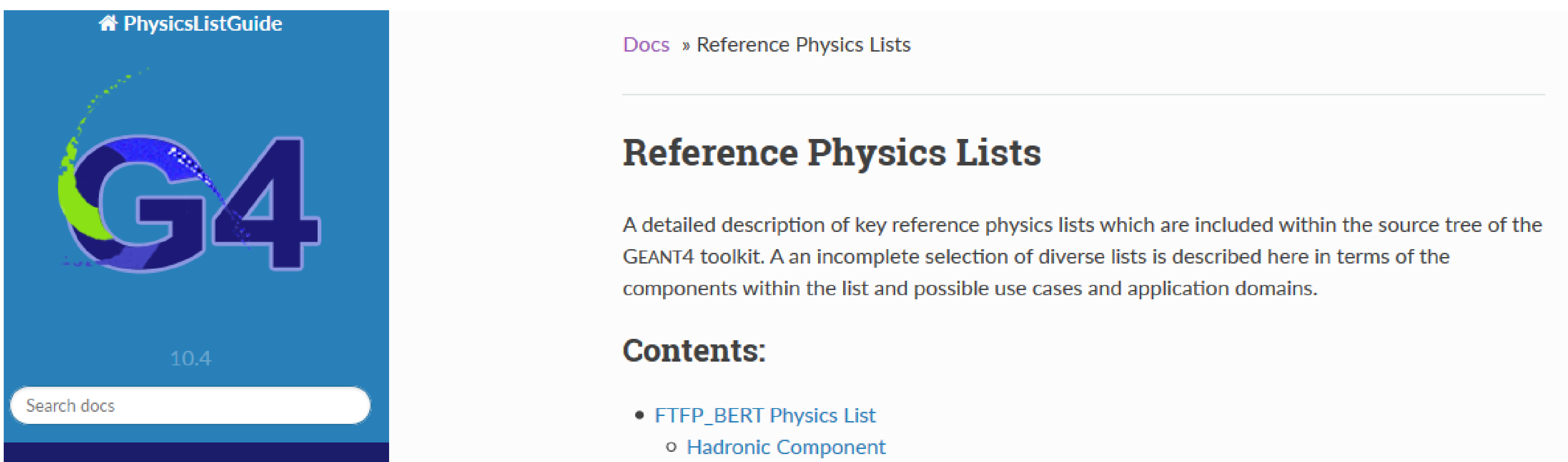
- Super-easy: in the **main()** function, just register an instance of the physics list to the **G4(MT)RunManager**:

```
#include "QGSP_BERT.hh"
int main() {
    // Run manager
    auto* runManager = G4RunManagerFactory::CreateRunManager();
    // ...
    G4VUserPhysicsList* physics = new QGSP_BERT();
    // Here, you can customize the "physics" object
    runManager->SetUserInitialization(physics);
    // ...
}
```

The complete lists of Reference Physics List

$G4INSTALL/source/physics_lists/lists

```
FTF_BIC.hh
FTFP_BERT.hh
FTFP_BERT_HP.hh
FTFP_BERT_TRV.hh
FTFP_INCLXX.hh
FTFP_INCLXX_HP.hh
G4GenericPhysicsList.hh
G4PhysListFactoryAlt.hh
G4PhysListFactory.hh
G4PhysListRegistry.hh
G4PhysListStamper.hh
INCLXXPhysicsListHelper.hh
LBE.hh
NuBeam.hh
QBBC.hh
QGS_BIC.hh
QGSP_BERT.hh
QGSP_BERT_HP.hh
QGSP_BIC_AllHP.hh
QGSP_BIC.hh
QGSP_BIC_HP.hh
QGSP_FTFP_BERT.hh
QGSP_INCLXX.hh
QGSP_INCLXX_HP.hh
Shielding.hh
```



Where to find information?

https://www.geant4.org/docs/

Geant4 Documentation

This page gives you an overview of all available documents which are created and maintained by the Geant4 international collaboration.

Introduction to Geant4

html - pdf - epub - kindle

This document gives you a more complete introduction to Geant4.

Installation Guide

html - pdf - epub - kindle

On this page

Summary – three kinds of physics lists for Geant4

- Old-style flat physics list
  - You code what you want, particle by particle and process by process
  - Very much flexible, but **not really encouraged**
- User-custom modular physics list
  - Blocks (constructors) provided by Geant4
  - Can register user-custom constructors
  - Usually the *optimal compromise* between flexibility and user-friendliness
- Ready-for-the-use Geant4 physics list
  - Plug and play (directly registered in the main!)
  - Can still register extra constructors

Part IV: Physics processes and models

Philosophy

- Provide a **general model framework** that allows the implementation of **complementary/alternative models** to **describe the same process** (e.g. Compton scattering)
  - A given model could work better in a certain energy range
- **Decouple** modeling of cross sections and of final state generation
- Provide **processes** containing
  - Many possible models and cross sections
  - Default cross sections for each model

Models under continuous development

Electromagnetic physics

Inventory (and specs) of the models for γ-rays

1 MeV γ in Al

- Many models available for each process
  - Plus one full set of polarized models
- Differ for energy range, precision and CPU speed
  - Final state generators
- Different mixtures available the Geant4 EM constructors

Model	E_{min}	E_{max}	CPU
G4LivermoreRayleighModel	100 eV	10 PeV	1.2
G4PenelopeRayleighModel	100 eV	10 GeV	0.9
G4KleinNishinaCompton	100 eV	10 TeV	1.4
G4KleinNishinaModel	100 eV	10 TeV	1.9
G4LivermoreComptonModel	100 eV	10 TeV	2.8
G4PenelopeComptonModel	10 keV	10 GeV	3.6
G4LowEPComptonModel	100 eV	20 MeV	3.9
G4BetheHeitlerModel	1.02 MeV	100 GeV	2.0
G4PairProductionRelModel	10 MeV	10 PeV	1.9
G4LivermoreGammaConversionModel	1.02 MeV	100 GeV	2.1
G4PenelopeGammaConversionModel	1.02 MeV	10 GeV	2.2
G4PEEFluoModel	1 keV	10 PeV	1
G4LivermorePhotoElectricModel	10 eV	10 PeV	1.1
G4PenelopePhotoElectricModel	10 eV	10 GeV	2.9

Similar situation for $e^{\pm}$

EM concept

- The same physics processes (e.g. Compton scattering) can be described by different models, that can be **alternative** or **complementary** in a given energy range
- For instance: Compton scattering can be described by
  - **G4KleinNishinaCompton**
  - **G4LivermoreComptonModel** (specialized low-energy, based on the Livermore database)
  - **G4PenelopeComptonModel** (specialized low-energy, based on the Penelope analytical model)
  - **G4LivermorePolarizedComptonModel** (specialized low-energy, Livermore database with polarization)
  - **G4PolarizedComptonModel** (Klein-Nishina with polarization)
  - **G4LowEPComptonModel** (full relativistic 3D simulation)
- Different models can be combined, so that the appropriate one is used in each given energy range (→ performance optimization)

When/why to use Low Energy Models

- **Use** Low-Energy models (Livermore or Penelope), as an *alternative* to Standard models, when you:
  - need precise treatment of EM showers and interactions at low-energy (keV scale)
  - are interested in atomic effects, as fluorescence x-rays, Doppler broadening, etc.
  - can afford a more CPU-intensive simulation
  - want to cross-check an other simulation (e.g. with a different model)
- Do **not** use when you are interested in EM physics > MeV
  - same results as Standard EM models, performance penalty

EM Physics Constructors for Geant4 10.4 - ready-for-the-use

G4EmStandardPhysics – default
G4EmStandardPhysics_option1 – HEP fast but not precise
G4EmStandardPhysics_option2 – Experimental
G4EmStandardPhysics_option3 – medical, space
G4EmStandardPhysics_option4 – optimal mixture for precision
G4EmLivermorePhysics
G4EmLivermorePolarizedPhysics
G4EmPenelopePhysics
G4EmLowEPPhysics
G4EmDNAPhysics_option…
…

Combined Physics
Standard > 1 GeV
LowEnergy < 1 GeV

- Advantage of using of these classes – they are tested on regular basis and are used for regular validation

Hadronic physics

(a very quick overview)

Hadronic Physics

- Data-driven models
- Parametrised models
- Theory-driven models

Hadronic physics challenge

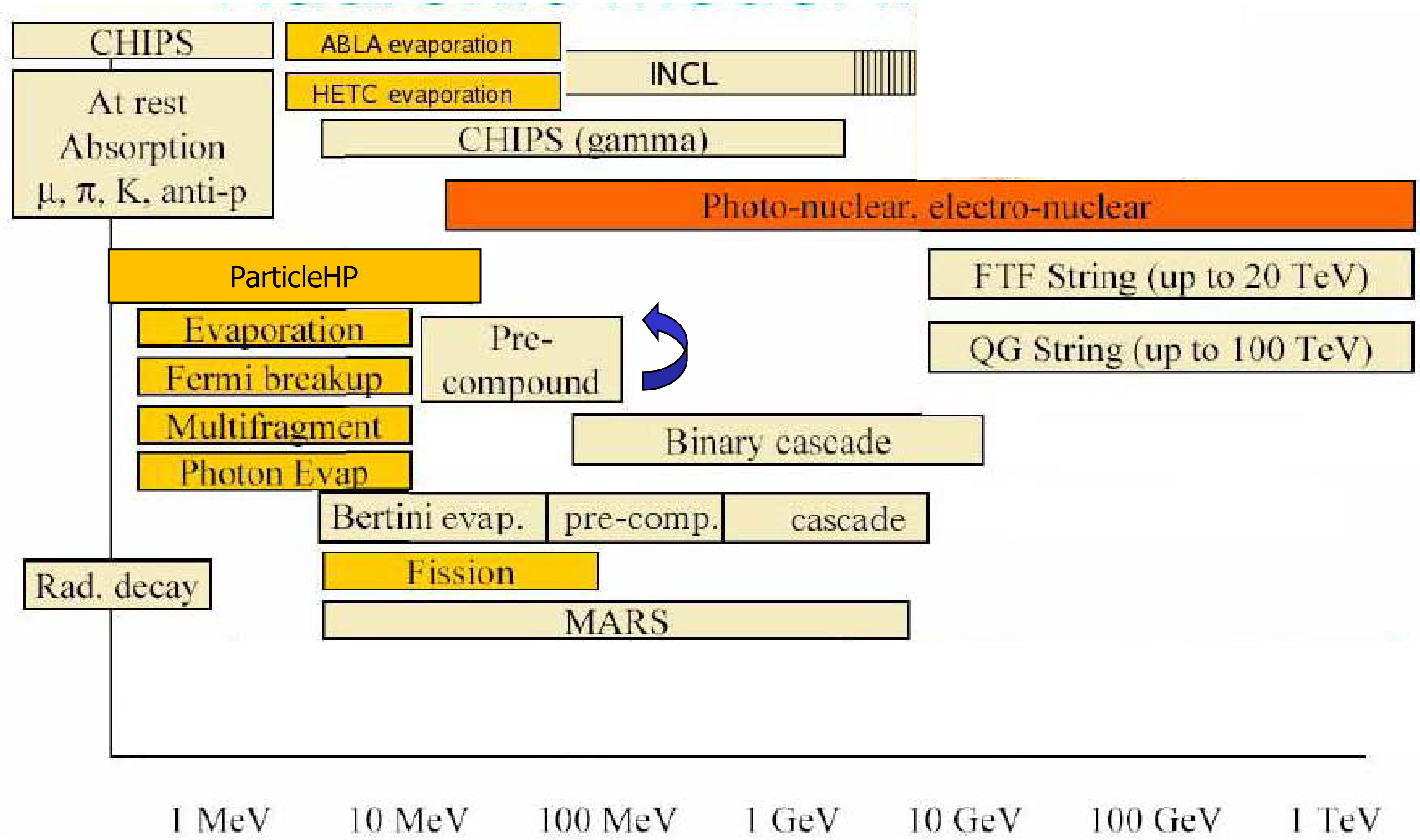
- Three energy regimes
  - < 100 MeV
  - resonance and cascade region (100 MeV - 10 GeV)
  - > 20 GeV (QCD strings)
- Within each regime there are several models
- Many of these are phenomenological

Reference physics lists for Hadronic interactions

- Two families of builders for the high-energy part (p, n, π and K)
  - **QGS**, or list based on a model that use the Quark Gluon String model for high energy hadronic interactions
  - **FTF**, based on the FTF (FRITIOF like string model)
- Three families for the cascade energy range
  - **BIC**, binary cascade
  - **BERT**, Bertini cascade
  - **INCLXX**, Liege Intranuclear cascade model
- "High precision" (**HP**) option, below 20 MeV
  - Database tracking for n, p, d, t, ^{3}He and α
  - Data from ENDFVII.r1 or TENDL-2014
  - CPU-thirsty

Hadronic model inventory

http://geant4.cern.ch/support/proc_mod_catalog/models



Hands-on session

- Task3
  - Task3a: Particles and processes
  - Task3b: Physics lists
  - Task3c: Production cuts

- **http://geant4.lns.infn.it/alghero2023/task3**