

---

# Pygate Documentation

*Release 1.0.0*

**vince**

**May 14, 2018**



---

## Contents:

---

|          |                                                     |           |
|----------|-----------------------------------------------------|-----------|
| <b>1</b> | <b>pygate introduction</b>                          | <b>3</b>  |
| 1.1      | An python interface to GATE . . . . .               | 3         |
| <b>2</b> | <b>Installation</b>                                 | <b>5</b>  |
| 2.1      | How to get pygate and configuration . . . . .       | 5         |
| <b>3</b> | <b>First step of pygate</b>                         | <b>7</b>  |
| 3.1      | 1. File <code>main.mac</code> generates. . . . .    | 7         |
| 3.2      | 2. Submit a task to subsystem . . . . .             | 7         |
| <b>4</b> | <b>pygate</b>                                       | <b>9</b>  |
| 4.1      | pygate introduction . . . . .                       | 9         |
| 4.2      | analysis . . . . .                                  | 9         |
| 4.3      | api . . . . .                                       | 9         |
| 4.4      | archive . . . . .                                   | 9         |
| 4.5      | components . . . . .                                | 10        |
| 4.6      | predefined . . . . .                                | 10        |
| 4.7      | routine . . . . .                                   | 11        |
| 4.8      | scripts . . . . .                                   | 11        |
| 4.9      | tests . . . . .                                     | 11        |
| 4.10     | utils . . . . .                                     | 12        |
| <b>5</b> | <b>API</b>                                          | <b>15</b> |
| 5.1      | for each file . . . . .                             | 15        |
| 5.2      | for function . . . . .                              | 15        |
| 5.3      | Here is the command list of <i>pygate</i> . . . . . | 38        |
| <b>6</b> | <b>Indices and tables</b>                           | <b>45</b> |



This is the master page of *pygate* documentation



## 1.1 An python interface to GATE

### GATE USER GUIDE

**Pygate** is a Subsystem based on python ,which runs with *GATE* simultaneously and speeds up *GATE* process. This Subsystem mainly contains two following functions,file generators and submitting experiments to Subsystem:

- file generators:
  - .mac file generator
  - .sh file generator
  - .C file generator
- submit system/experiment run system:
  - dependency system:
    - \* via slurm
    - \* easy constructing task train

*ref-1. File main.mac generates. ref-2. Submit a task to subsystem*

If you want more information,you can go to [github](#) to get Pygate.

More efforts needs to be made to perfect \*pygate\*.  
We warmly welcome whom (now just **for** external members)  
are interested **in** this program to comeforward **and** find  
out more about us,join us **and** make full use of this platfom.

predefined *First step of pygate*



### 2.1 How to get pygate and configuration

- if you want to run *pygate* locally, you should follow the steps:
  1. We put the source on the [Github](#). You may need a Github account to clone or download the files. Here is the [Github Guides](#).
  2. **Ensure your python3 version is the most current version.**
    - We recommend *Anaconda* to get *python3*. Here is the [Anaconda](#).
    - The *Anaconda* should be set into `syspath`.
    - `source ~/.bashrc` or reboot the terminal to update `bashrc`.
    - `$ python --version` and get the output `Python 3.6.4 :: Anaconda custom(64-bit) (for now)`.
  3. Install *pygate*
    - `pip install dsl-pygate`
  4. Ensure *GATE* is installed and configured already.
  5. You have already installed *pygate*. Go to the [First step of pygate](#).
- if you are an external member (you haven't gotten an account), you can run on the server:
  1. Install *Anaconda* to get latest *python3* in your work folder. The recommended path with high performance is `/mnt/Gluster_NoGPU/usr`.
  2. **Ensure your python3 version is the most current version.**
    - We recommend *Anaconda* to get *python3*. Here is the [Anaconda](#).
    - The *Anaconda* should be set into `syspath`.
    - `source ~/.bashrc` or reboot the terminal to update `bashrc`.

```
- $ python --version and get the output Python 3.6.4 :: Anaconda  
custom(64-bit) (for now).
```

3. Install *pygate*

```
- pip install dsl-pygate
```

4. **Configured GATE**

```
- source source/hqlf/softwarewares/moudle/simu8.0.sh
```

5. You have made *pygate* ready. Go to the [First step of pygate](#).

---

**Note:** We will get the environment set up and configured on each node of the server. You need to install and configure the environment in your own work folder at present.

---

[Github Anaconda](#)

---

### First step of pygate

---

In this page, we will run an example consisting of two steps;

#### 3.1 1. File `main.mac` generates.

- Before a *GATE* process, we generally need to make a file of `main.mac`, in which we can set the components ( *world, system, phantom, source and digitizer etc.* ) for *GATE* simulation.
- The question is that the definition of components is too heavy and complicated. A lot of time and energy wastes on this.
- *Pygate* offers a function of **File Generator**. Users can configure the components easily by setting several arguments of necessary components you want in a file of `make_mac.py` (you can name it freely). A file of `mac.yml` may be needed containing some default settings of the `main.mac`, or you can set these in `make_mac.py` directly.
- **The command for `main.mac` generating is:**
  - `$ pygate generate mac script -t make_mac.py -o main.mac -c mac.yml.`
  - `make_mac.py` and `mac.yml` should be included in current work folder.
  - Then you will find the `main.mac` in the folder.
- The main work for this step is to code `make_mac.py`. Users can modify on the template for first time.

:ref:make\_mac.py

#### 3.2 2. Submit a task to subsystem

- Usually there are hundreds of millions, even billions of events occurring during a *GATE* simulation, which is the reason why the process takes a long time.

- However, it is a repeatable work for a *GATE* program to generate an event. *Pygate* offers a method to speed up the process. *Pygate* divides the task into a lot of parts,
- then submits these parts to server. Each part of the original task will be distributed to no-working machine of the net by *SLurm*. Thus, we get a very high speed for *GATE* simulation.

- **Users should know the following steps to archive it:**

1. **Users should get the needed configured files by executing this command: `$ pygate init ext`. You will get the**

- `main.mac`, you get it last in the folder.
- `GateMaterials.db`, significant file for *GATE* configuration, can't be lack.
- `Hits2CSV.C`, may needed if you want the data of *csv* format.
- `Materials.xml`
- `Surface.xml`, set the surface rendering. Or you can select volume rendering.

2. **When you get the necessary files in the work folder, you need to divide the task into parts.**

- `$ pygate init subdir -n --INTEGER -f --STR`, you can set the number of parts and the name of subdirectories as you want. The default option is "sub.[10]" and you will get 10 subdirectories of "sub.[x]" (x~[0-10]).
- `$ pygate init bcast`, broadcast the files to subdirectories made last step.
- `$ pygate generate shell`, generate `run.sh` for *SLurm* to distribute the task and `post.sh` to merge the results of each parts.
- `$ pygate submit`, submit the task to subsystem. *SLurm* will do the distribution. The details information of distribution will print on the screen. You can easily know which machine each part run.
- **There are two procedures before getting results:**
  - \* First, the machines absorb the mission and complete it, then feedback the results to subdirectories. `run.sh` is for this step.
  - \* Then the results from subdirectories are merged into one file of `optical.root`, containing all collected data of Hits. `post.sh` is for this.

You can refer the detail of commands in [Here is the command list of pygate](#)

*pygate docomentions' index*

## 4.1 pygate introduction

## 4.2 analysis

- `_init_.py`
- `results.py`

## 4.3 api

- **cli**
  - `_init_.py`
  - `base.py`
  - `commands.py`
- `_init_.py`

## 4.4 archive

- **macs**
  - `mct2d_source.mac`
- `_init_.py`

- mac\_templates.yml
- maxdepth\_bash\_sample.sh
- map\_bash.sh
- map\_zsh.sh
- merge\_bash.sh
- merge\_bash\_zsh.sh
- pygate.yml

## 4.5 components

- **geometry**
  - **camera**
    - \* \_init\_.py
    - \* camera.py
    - \* system.py
  - \_init\_.py
  - geometry.py
  - phantom.py
  - surface.py
  - volume.py
- templates
- \_init\_.py
- base.py
- digitizer.py
- misc.py
- parameter.py
- physics.py
- simulation.py
- source.py
- utils.py

## 4.6 predefined

- \_init\_.py
- \_camera.py
- \_sources.py
- cameras.py

- digitizers.py
- parameters.py
- phantoms.py
- physice.py
- simulations.py
- source.py

## 4.7 routine

- \_init\_.py
- analysis.py
- base.py
- cleaner.py
- initialize.py
- merger.py
- submit.py
- utils.py

## 4.8 scripts

- templates
- \_init\_.py
- base.py
- helper.py
- shell.py

## 4.9 tests

- components
- predefined
- routine
- scripts
- \_init\_.py
- test\_methods.py
- test\_shell.py

## 4.10 utils

- `_init_.py`
- `object_with_template.py`
- `strs.py`
- `typing.py`

pygate

- **analysis**
  - `_init_.py`
  - `results.py`
- **api**
  - **cli**
    - \* `_init_.py`
    - \* `base.py`
    - \* `commands.py`
  - `_init_.py`
- **archive**
  - **macs**
    - \* `mct2d_source.mac`
  - `_init_.py`
  - `mac_templates.yml`
  - `maxdepth_bash_sample.sh`
  - `map_bash.sh`
  - `map_zsh.sh`
  - `merge_bash.sh`
  - `merge_bash_zsh.sh`
  - `pygate.yml`
- **componets**
  - **geometry**
    - \* **camera** `_init_.py` `camera.py` `system.py`
    - \* `_init_.py`
    - \* `geometry.py`
    - \* `phantom.py`
    - \* `surface.py`
    - \* `volume.py`
  - `templates`

- `_init_.py`
- `base.py`
- `digitizer.py`
- `misc.py`
- `parameter.py`
- `physics.py`
- `simulation.py`
- `source.py`
- `utils.py`

- **predefined**

- `_init_.py`
- `_camera.py`
- `_sources.py`
- `cameras.py`
- `digitizers.py`
- `parameters.py`
- `phantoms.py`
- `physice.py`
- `simulations.py`
- `source.py`

- **routine**

- `_init_.py`
- `analysis.py`
- `base.py`
- `cleaner.py`
- `initialize.py`
- `merger.py`
- `submit.py`
- `utils.py`

- **scripts**

- `templates`
- `_init_.py`
- `base.py`
- `helper.py`
- `shell.py`

- **tests**

- components
  - predefined
  - routine
  - scripts
  - `_init_.py`
  - `test_methods.py`
  - `test_shell.py`
- **utils**
  - `_init_.py`
  - `object_with_template.py`
  - `strs.py`
  - `typing.py`
- `_init_.py`
- `cleaner.py`
- `config_maker.py`
- `config.py`
- `configs.py`
- `initializer.py`
- `merger.py`
- `phantom.py`
- `renderable.py`
- `service.py`
- `shell.py`
- `submitter.py`
- `utils.py`

### **5.1 for each file**

#### **5.1.1 analysis**

#### **5.1.2 api**

#### **5.1.3 archive**

#### **5.1.4 components**

#### **5.1.5 predefined**

#### **5.1.6 routine**

#### **5.1.7 scripts**

#### **5.1.8 tests**

#### **5.1.9 utils**

### **5.2 for function**

#### **5.2.1 cli**

cli

### main.py

- auto\_sub()
- load\_config(filename, is\_no\_config, dryrun)
- pygate(config, no\_config, dryrun)

### analysis.py

- analysis\_kernel(source, target, analysis\_type, dryrun)
- predefined(name, source, output)
- script(target, source, output):

### clean.py

- **clean\_kernel(is\_subdirectories, subdirectory\_patterns: Iterable[str], root\_file\_patterns: Iterable[str], is\_slurm\_outputs, dryrun: bool)**
- clean(subdirectories, root\_files, slurm\_outputs)

### initialize.py

- init
- **generate**
  - mac\_template(filename)
  - shell\_task\_list(tasks)
  - shell\_run(filename,tasks,gate\_version,shell,partition)
  - shell\_post\_run(filename,tasks,gate\_version,shell,partition)
  - shell\_post\_run(filename,tasks,gate\_version,shell,partition)
  - shell()
  - cfg(target,format)
  - subdir(nb\_split,sub\_format)
  - broadcast\_kernel(files,subdirectory\_patterns,dryrun)
  - bcast(target, no\_ext)
  - external\_to\_copy()
  - ext()
  - auto()

### mac\_generate.py

- **mac()**
  - script()
  - predefined()

## merge.py

- **Tasks**
  - `__init__(self,filename,method)`
- `merge_kernel(tasks:Iterable[Taks], subdir_pattern: Tterable[str],dryrun)`
- `merge(target, method)`

## submit.py

- **Task**
  - `__init__(self, broadcast=None,single=None)`
- `submit_kernel(taks:Iterable[Task],subdir_patterns:Iterable[str],duyrun)`
- `submit(broadcast,single)`

## 5.2.2 analysis

analysis

### `__init__.py`

### `predifined.py`

- `gamma_energy_deposite_distribution(csv_filename, h5_filename)`

## results.py

- `ParticleID(Enum)`
- `ColumnNames`
- **ResultBase**
  - `__init__(self,data=None)`
  - `d(self)`
- **Results(Resultbase)**
  - `__init__(self,data:Tuple[ResultBase])`
  - `map(self,func) -> 'Results'`
  - `fileter(self,func) -> 'Results'`
  - `call(self, fuc_name) -> 'Results'`
  - `zip(self, r: 'Results')`
  - `first(self) -> ResultBase`
  - `flatten(self) -> 'Results'`
  - `merge(self) ->ResultBase`
  - `to_list(self)`

- **ResultsDask(ResultBase)**
- **ResultsNamedTuple(ResultBase)**
- **ResultsWithKeys(Results)**
  - `__init__(self, data: Tuple[Tuple[str, ResultBase]])`
  - `drop_keys(self) -> Results`
  - `to_dict(self)`
  - `select(self, key)`
- **ResultsWithUnknownKeys(ResultsDask)**
  - `select(self, key)`
  - `drop_keys(self) -> ResultsDask`
- **Series(ResultBase)**
  - `__init__(self, series: pd.Series)`
  - `columns(self, *cols)`
  - `position(self) -> 'Vec3'`
  - `source_position(self) -> 'Vec3'`
  - `energy_deposit(self) -> float`
- **DataFrame(ResultBase)**
  - `__init__(self, dataframe: pd.DataFrame)`
  - `first(self, *columns) -> Series`
  - `split_row(self) -> Results`
  - `merge(self, r) -> 'DataFrame'`
  - `to_event(self) -> 'Event'`
- **Vec3(ResultBase)**
  - `__init__(self, x, y=Mone, z=None)`
  - `x(self) -> float`
  - `y(self) -> float`
  - `z(self) -> float`
  - `to_list(self)`
- **EnergyDeposit(ResultBase)**
  - `def __init__(self, position, energy)`
  - `position(self)`
  - `energy(self)`
- **Event(DataFrame)**
  - `__init__(self, dataframe: pd.DataFrame)`
  - `source_position(self) -> Vec3`
  - `first_position(self) -> Vec3`

- incident\_direction(self) -> Vec3
  - energy\_deposit\_list(self) -> Results
- **CSVFile(ResultBase)**
  - \_\_init\_\_(self,filename:str)
  - load(self) -> DataFrame

### 5.2.3 api

api

cli

\_\_init\_\_.py

---

### base.py

- **CLI(click.MultiCommand)**
  - \_\_init\_\_(self)
  - list\_commands(self,ctx)
  - get\_command(self, ctx, name)

### commands.py

- \_load\_config(config)
- make\_config()
- init(config,content)
- submit(config)
- merge(config)
- clean(config, content, dryrun)

### 5.2.4 archive

archive

### 5.2.5 components

components

## geometry

### geometry.py

- **GeometryPreInitialise(ObjectWithTemplate)**
  - `__init__(self, world:Volume)`
- **GeometryPostInitialise(ObjectWithTemplate)**
  - `__init__(self, surfaces)`
- **Geometry**
  - `__init__(self, world:Volume)`
  - `render_pre(self)`
  - `render_post(self)`

### phantom.py

- **Phantom(ObjectWithTemplate)**
  - `__init__(self, sensitive_detectors: Tuple[Volume]=())`

### surface.py

- **Surface(ObjectWithTemplate)**
  - `__init__(self, base: Volume, insert:Volume)`
- **SurfaceOerfectAPD(Surface)**
- **SurfaceRoughTeflonWrapped(Surface)**

### volume.py

- **Repeater(ObjectWithTemplate)**
- **RepeaterRing(Repeater)**
  - `__init__(self, number)`
- **RepeaterLinear(Repeater)**
  - `__init__(self, number, repeat_vector)`
- **RepeaterCubic(Repeater)**
  - `__init__(self, scale: Vec3, repeat_vector: Vec3)`
- **Volume(ObjectWithTemplate)**
  - `__init__(self, name, material=None, mother=None, position=None, unit=None, repeaters: Repeater=None)`
  - `add_child(self, child)`
- **Box(Volume)**

- `__init__(self, name, size, material=None, ,other=None, position=None,unit=None,repeaters: Repeater=None)`
- **Cylinder(Volume)**
  - `__init__(self, name, rmax, rmin=None, height=None,phi_start=None,delta_phi=None,material=None,mother=None,Repeater=None)`
- **Sphere(Volume)**
  - `__init__(self, name, rmax, rmin=None, phi_start=None,delta_phi=None,theta_start=None,delta_theta=None,material=None)`
- **ImageRegularParameterisedVolume(Volume)**
  - `__init__(self, name, image_file, range_file, material=None,mother=None,position=None,init=None,repeaters: Repeater=None)`
- **Patch(Volume)**
  - `__init__(self, name, patch_file, material=None,mmother=None,position=None, unit=None, repeater:Repeater=None)`

## camera

### `__init__.py`

### `camera.py`

- **Camera(ObjectWithTemplate)**
  - `__init__(self, system:System, sensitive_detectors:Tuple[Volume]=())`

## system.py

- **System**
  - `__init__(self)`
- **PETscanner(System)**
  - `__init__(self,level1, level2, level3, level4, level5, sensitive_detectors=None)`
- **Ecat(System)**
  - `__init__(self, block=None, crystal=None)`
- **CylindricalPET(System)**
  - `__init__(self, rsector=None, module=None,submodule=None,crystal=None,layer0=None,layer1=None,layer2=None,`
- **MultiPatchPET(System)**
  - `__init__(self, container, patch_list)`
- **SPECThead(System)**
  - `__init__(self,crystal,pixel=None)`
- **OpticalSystem(System)**
  - `__init__(self, crystal,pixel = None)`

## template

### digitizer

- **singles**
  - blurring.j2
  - buffer.j2
  - dead\_time.j2
  - holder.j2
  - readout.j2
  - singles.j2
- co\_incidence\_chain.j2
- co\_incidence\_sorter.j2
- insertable.j2

### geometry

- **volume**
  - **repeater**
    - \* cubic.j2
    - \* linear.j2
    - \* repeater.j2
    - \* ring.j2
  - box.j2
  - cylinder.j2
  - image\_sphere.j2
  - sphere.j2
  - volume.j2
- camera.j2
- geometry\_post\_init.j2
- geometry\_pre\_init.j2
- phantom.j2
- surface.j2
- system.j2

## **misc**

- database.j2
- verbose.j2
- visualisation.j2

## **parameter**

- **aquisition**
  - aquisition.j2
  - period.j2
  - primaries.j2
- **output**
  - output.j2
  - root.j2
  - sinogram.j2
- **random\_engine**
  - random\_engine.j2
- parameter.j2

## **physics**

- cuts.j2
- list.j2
- model.j2
- physics.j2
- procrss.j2

## **source**

- **angular**
  - angular.j2
  - iso.j2
- **particle**
  - gamma.j2
  - particle
  - position.j2
- **shape**
  - annulus.j2

- circle.j2
- cylinder.j2
- ellipse.j2
- ellipsoid.j2
- rectangle.j2
- shape.j2
- sphere.j2
- voxelized.j2
- source\_list.j2
- source.j2

## simulation.j2

## test.j2

## utils.j2

## vec3.j2

## \_\_init\_\_.py

## base.py

- ObjectWithTemplate(ObjectWithTemplateBase)
- **Renderable**
  - render(self) -> str

## digitizer.py

- **Insetable(ObjectWithTemplate)**
  - \_\_init\_\_(self, name=None, is\_define\_name=False, is\_explicit\_insert=True)
- **Singles(Insertable)**
  - \_\_init\_\_(self, plugins=None, name='Singles', is\_define\_name=False, is\_explicit\_insert=False)
- **AdderCompton(Insetable)**
  - \_\_init\_\_(self, name='adderCompton', is\_define\_name=False)
- **Readout(Insertable)**
  - \_\_init\_\_(self, policy=None, depth=1, name='readout', is\_define\_name=False)
- **Blurring(Insertable)**
  - \_\_init\_\_(self, law=None, resolution=0.15, eor=511, slope=None, name='blurring', is\_define\_name=False)
- **Holder(Insetable)**

- `__init__(self, value, name=None, is_define_name=False)`
- **ThresHolder(Holder)**
  - `__init__(self, value, name='thresHolder')`
- **UpHolder(Holder)**
  - `__init__(self, value, name='upholder')`
- **TimeResolution(Insetable)**
  - `__init__(self, resolution, name=None, is_define_name=False)`
- **WithBuffer(Insertable)**
  - `__init__(self, size=None, mode=None, name=None, is_define_name=False)`
- **MemoryBuffer(WithBuffer)**
  - `__init__(self, read_freq=None, size=None, mode=None, name=None, is_define_name=False)`
- **DeadTimeMulti(WithBuffer)**
  - `__init__(self, volume, t, mode=None, buffer_size=None, buffer_mode=None, name='deadtime', is_define_name=False)`
- **SingleChain(Singles)**
  - `__init__(self, name, is_define_name=True)`
- **CoincidenceSorter(Insertable)**
  - `__init__(self, input_=None, window=None, offset=None, name='Coincidences', is_define_name=False, is_explicit_insert=False)`
- **CoincidencesChanin(Insertable)**
  - `__ini__(self, input1, input2, name, plugins=None, use_priority=None, conseve_all_event=None, is_define_name=True)`

## misc.py

- **Verbose(ObjectWithTemplate)**
  - `__init__(self, physics=0, cuts=0, sd=0, action2=0, actor=0, step=0, error=0, warning=0, output=0, beam=0, volume=0, imag`
- **MateriaDatabase(ObjectWithTemplate)**
  - `__init__(self, path: str=None)`
- **MaterialDatabaseLocal(MaterialDatabase)**
  - `__init__(self)`
- **Visualisation(ObjectWithTemplate)**
  - `__init__(self, is_disable=True)`

## parameter.py

- **Acquisition(ObjectWithTemplate)**
- **AcquisitionPrimaries(Acquisition)**
  - `__init__(self, number=10000)`

- **AcquisitionPeriod(Acquisition)**
  - `__init__(self, start=0.0, end=1.0, step=1.0)`
- **Output(ObjectWithTemplate)**
  - `__init__(self, file_name)`
- **Ascii(Output)**
  - `__init__(self, file_name, hit=0, singles=0, coincidences=0)`
- **Binary(Output)**
  - `__init__(self, file_name, hit=0, singles=0, coincidence=0)`
- **Root(Output)**
  - `__init__(self, file_name, hit=None, singles=None, coincidences=None, optical=None, delay=None)`
- **Sinogram(Output)**
  - `__init__(self, file_name, input_, radial_bin=None, is_true_only=None, is_raw_output=None, tang_blurring=None, is_sto)`
- **RandomEngine(ObjectWithTemplate)**
  - `__init__(self, seed='default')`
- **RandomEngineRanlux64(RandomEngine)**
- **RandomEnginJamesRandom(RandomEngine)**
- **RandomEngineMersenneTwister(RandomEngine)**
- **Parameter(ObjectWithTemplate)**
  - `__init__(self, random_engine, outputs: List[output], acquisition)`

## physics.py

- **Model(ObjectWithTemplate)**
  - `__init__(self, particle=None)`
- **PenelopeModel(Model)**
- **StandarModel(Model)**
- **LivermoreModer(Model)**
- **LivermorePolarizeModel(Model)**
- **PhysicsProcess(ObjectWithTemplate)**
  - `__init__(self, models=None)`
  - `content_in_adding(self)`
- **PhotoElectric(PhysicsProcess)**
- **Compton(PhysicProcess)**
- **GammaConversion(PhysicsProcess)**
- **RayleighScattering(PhysicsProcess)**
  - `__init__(self, models=(PenelopeModel(),))`
- **ElectronIonisation(PhysicsProcess)**

- `__init__(self, models=(StandardModel('e-'),StandardModel('e+')))`
- **Bremsstrahlung(PhysicsProcess)**
  - `__init__(self,models=(StandardModel('e-'),StandardModel('e+')))`
- **PhysicsProcessWithoutModels(PhysicsProcess)**
  - `__init__(self)`
- **PositronAnnihilation(PhysicsProcessWithoutModels)**
- **RadioactiveDecay(PhysicsProcessWithoutModels)**
- **OpticalAbsorption(PhysicsProcessWithoutModels)**
- **OpticalRayleigh(PhysicsProcessWithoutModels)**
- **OpticalBoundary(PhysicsProcessWithoutModels)**
- **OpticalMie(PhysicsProcessWithoutModels)**
- **OpticalWLS(PhysicsProcessWithoutModels)**
- **Scintillation(PhysicsProcessWithoutModels)**
- **MultipleScattering(PhysicsProcessWithoutModels)**
  - `__init__(self, particle)`
  - `content_in_adding(self)`
- **EMultipleScattering(PhysicsProcessWithoutModels)**
  - `__init__(self,particle)`
  - `content_in_adding(self)`
- **PhysicsList(ObjectWithTemplate)**
  - `__init__(self,physics_processes:List[PhysicsProcess])`
- **Cuts(ObjectWithTemplate)**
  - `__init__(self,volume:Volume,cuts:TV('CutT',float,Dict[Volume,float]),max_step: float=None)`
- **Physics(ObjectWithTemplate)**
  - `__init__(self, physics_list,cut_list)`

## simulation.py

- **Simulation(ObjectWithTemplate)**
  - `__init__(self,geometry,physics,digitizers,source,paramerter,material_database=None,visualisation=None)`

## source.py

- **Source(ObjectWithTemplate)**
  - `__init__(self,name,particle=None,activity=None,angle=None,shape=None,position: Vec3=None)`
  - `unified_activity(self, activity)`
  - `bind(self, obj)`
  - `is_voxelized(self)`

- **Particle(ObjectWithTemplate)**
  - `bind_source(self, source)`
  - `__init__(self, unstable=None, halflife=None)`
- **ParticlePositron(Particle)**
  - `__init__(self, unstable=True, halflife=6586)`
- **ParticleGamma(Particle)**
  - `__init__(self, unstable=True, halflife=6586.2, monoenergy=511, back2back=True)`
- **Angular(ObjectWithTemplate)**
- **Shape(ObjectWithTemplate)**
  - `__init__(self, dimension)`
- **ShapePlane(Shape)**
  - `__init__(self)`
- **ShapeSurfaceOrVolume(Shape)**
  - `__init__(self, dimension)`
- **Voxelized(Shape)**
  - `__init__(self, read_table, read_file, reader='interfile', translator='range', positon=None)`
- **Cylinder(ShapeSurfaceOrVolume)**
  - `__init__(self, radius, halfz, dimension)`
- **Sphere(Shape)**
  - `__init__(self, radius, dimension)`
- **Ellipsoid(Shape)**
  - `__init__(self, half_size: Vec3, dimension)`
- **Circle(ShapePlane)**
  - `__init__(self, radius)`
- **Annulus(ShapePlane)**
  - `__init__(self, radius0, radius)`
- **Ellipse(ShapePlane)**
  - `__init__(self, half_size)`
- **Rectangle(ShapePlane)**
  - `__init__(self, half_size)`
- **SourceList(ObjectWithTemplate)**
  - `__init__(self, sources: List[Source])`

## utils.py

- **Vec3(ObjectWithTemplate)**
  - `__init__(self, x, y, z, unit=None)`

### 5.2.6 predefined

- `cylindricalPET(world: Volume, cylinder=None, rrh=None, head=None, rcb=None, block=None, rcc=None, crystal=None, lso=None, lsc=None)`
- `ecat(world: Volume, cylinder=None, rlb=None, rrb=None, block=None, rcc=None, crystal=None)`
- `opticalsystem(world: Volume, box=None, crystal=None, rcp=None, pixel=None)`
- `optical_gamma(world: Volume, crystal: VolumeLike)`
- `multipatchPET(world: Volume)`
- `optocal_surfaces(cam: Camera)`
- `voxelized_gamma(position, src_name='voxelized_gamma', read_table='act_range.data', read_file='act.h33')`
- `voxelized_F18(position, src_name='voxelized_F18', read_table='act_range.dat', read_file='act.h33')`
- `cylinder_source(position=Vec3(0,0,0), src_name='cylinder_source', cylinder=None, activity=None, particle=None, angle=None)`
- `plane_source(position=Vec3(0,0,0), src_name='plane_source', rectangle=None, activity=None, particle=None, angle=None)`
- `sphere_source(position=Vec3(0,0,0), src_name='sphere_source', sphere=None, activity=None, particle=None, angle=None)`
- `make_default_source(simu_name)`
- `sphere(radius, position: Vec3=Vec3(0.0,0.0,0.0,'mm'), angle: Anular=None, activity: int=1000, particle=None, name='sphere_source', -> SourceList`
- `ecat_digitizer(dtvolume, rdr=None, blur=None, thres=None, uph=None, ddt=None, coin=None, coin_delay=None, coin_chain=None)`
- `cylindricalPET_digitizer(rdr=None, blur=None, thres=None, uph=None, coin=None, coin_delay=None)`
- `optical_digitizer(rdr=None)`
- `spect_digitizer(blur=None, spblur=None, thres=None, uph=None)`
- `pet_parameters(acqu=AcquisitionPeriod(), outputs=None, rand=RandomEngine(seed='auto'))`
- `optical_parameters(acqu=AcquisitionPrimaries(number=10000), outputs=None, rand=RandomEngineMersenneTwister(seed='auto'))`
- `optical_gamma(nb primaries)`
- `voxelized_phantom(world, image_file='phan.h33', range_file='mat_range.dat', position=None)`
- `pet_physics(cut_pair_list)`
- `optical_physics(cut_pari_list)`
- `gamma_physics(cut_pair_list)`
- `spect_physics(cut_pair_list)`
- `PredefinedSimulations(Enum)`
- `make_default_camera(simu_name, world: Volume)`
- `make_default_surfaces(simu_name, cam: Camera)`
- `make_default_physics(simu_name, cam: Camera, phan: Phantom, cut_pair_list=None)`
- `make_default_digitizer(simu_name, cam: Camera)`
- `make_default_parameter(simu_name)`
- `make_simulation(simu_name, geo=None, phy=None, digi=None, src=None, para=None)`
- `simulation(simulation_name: PredefinedSimulations, geometry=None, physics=None, digitizer=None, source=None, parameter=None)`
- `optical_gamma(source=None, world_size: Vec3=Vec3(400.0,400.0,400.0,'cm'), crystal_size: Vec3=Vec3(30.0,30.0,30.0,'mm'), crys`

## 5.2.7 routine

routine

- KEYS
- AnalysisType(Enum)
- **OperationAnalysis(OperationOnfile)**
  - `__init__(self, source_csv_filename, target_h5_filename, analysis_type:AnalysisType=None)`
  - `source(self,r:RoutineOnDirectory)`
  - `dryrun(self,r:RoutineOnDirectory)`
- **Operation**
  - `apply(self,routine)`
  - `dryrun(self,routine)`
- **Routine**
  - `__init__(self,operation:TIterable[Operation]=(),dryrun=False,verbose=0)`
  - `last_result(self)`
  - `work(self)`
  - `echo(self)`
- RoutineOnDirectory(Routine)
- **OperationOnFile(Operation)**
  - `__init__(self,filename:str)`
  - `target(self,r:RoutineOnDirectory) -> File`
  - `dryrun(self,r:RoutineOnDirectory)->Dict[str,Any]`
  - `apply(self,r: RoutineOnDirectory) ->Dict[str,Any]`
- **OperationOnSubdirectories(Operation)**
  - `__init__(self,patterns:Iterable[str])`
  - `subdirectories(self, r: RoutineOnDirectory)`
- **OperationWithShellCall(Operation)**
  - `call_args(self,r:Routine)`
  - `stdout(self, r:Routine)`
  - `run_child_program(self, r:Routine)`
  - `apply(self,r:Routine)`
  - `dryrun(self, r:Routine)->Dict[str, Iterable[str]]`
- KEYS
- **OpCleanSubdirectories(OperationOnSubdirectories)**
  - `apply(self, r: RoutineOnDirectory)`
  - `dryrun(self, r:RoutineOnDirectory)`
- **OpCleanSource(Operation)**

- `__init__(self, file_patterns: Iterable[str])`
- `files(self, r: RoutineOnDirectory)`
- `apply(self, r: RoutineInDirectory)`
- `dryrun(self, r: RoutineOnDirectory)`
- **OpCleanFilesInAllDirectories(OperationOnSubdirectories)**
  - `__init__(self, file_patterns: Iterable[str], subdirectory_patterns: Iterable[str])`
  - `files_on_directory(self, d: Directory) need to be corrected(directroy->directory)`
  - `to_remove(self, r: RoutineOnDirectory)`
  - `apply(self, r: RoutineOnDirectory)`
  - `dryrun(self, r: RoutineOnDirectory)`
- `routine_clean(source_patterns: Iterable[str]=(), is_subdir: bool=False, dryrun: bool=False)`
- KEYS
- **TargetFileWithContent**
  - `__init__(self, target: TypeVar('FileLike', File, str).is_to_broadcast: bool=True, content: str=None)`
  - `to_dict(self)`
  - `save(self)`
- **OpAddToBroadcastFile(OperationOnFile)**
  - `__init__(self, filename)`
  - `apply(self, r: RoutineOnDirectory)`
  - `dryrun(self, r: RoutineOnDirectory)`
- **OpGenerateFile(OperationOnFile)**
  - `__init__(self, filename: str, is_to_broadcast=True)`
  - `content(self, r: RoutineOnDirectory) -> str`
  - `target_file_with_content(self, r: RoutineOnDirectory)`
  - `apply(self, r: RoutineOnDirectory)`
  - `dryrun(self, r: RoutineOnDirectory)`
- **OpGeneratorMac(OpGenerateFile, OpeartionWithShellCall)**
  - `__init__(self, script_filename: str, mac_config: str=None, mac_filename: str=None)`
  - `call_args(self, r: RoutineOnDirectory)`
- **OpgenerateMacTemplate(OpGerateFile)**
  - `__init__(self, filename: str, script: Script)`
  - `content(self, r: RoutineOnDirectory) -> str`
- **OpGeneratorShell(OperationOnFile)**
  - `__init__(self, filename: str, script: Script)`
  - `content(self, r: RoutineOnDirectory) ->`
- **OpGeneratorPhantom(OperationOnFile)**

- `__init__(self,filename:str)`
- **OpSubdirectoriesMake(Operation)**
  - `__init__(self,nb_split:int,subdirectory_format:str="sub.{ }")`
  - `apply(self, r: RoutineOnDirectory)`
  - `dryrun(self, r: RoutineOnDirectory)`
- **OpBroadcastFile(OperationONsubdirectories)**
  - `files_to_broadcast(self,r: RoutineOnDirectory)`
  - `dryrun(self,r:RoutineOnDirectory)`
  - `apply(self,r: RoutineOndirectory)`
- **OpMerge(OperationOnFile,OperationOnSubdirectories)**
  - `__init__(self,filename:str,patterns:Iterable[str])`
  - `sources(self, r:RoutineOnDirectory)`
  - `dryrun(self,r: RoutineOnDirectory) -> JSONStr`
- **OpMergeWithShellCall(OpMerge,OpeartionWithShellCall)**
  - `__init__(self,filename:str,patterns:Iterable[str])`
  - `apply(self,r:RoutineOnDirectory)`
  - `dryrun(self,r:RoutineOnDirectory)`
- **OpMergeHADD(OpMergeWithShellCall)**
  - `call_args(self,r: RoutinOnDirectory)`
- **OpMergeCat(OpMergeWithShellCall)**
  - `call_args(self,r:RoutineOnDirectory)`
- **OpMergePandasConcatenate(OpMerge)**
  - `apply(self,r:RoutineOnDirectory)`
- **OpMERgeSumBinary(OpMerge)**
- `hadd(work_derectory:Directory,subdirectory_patterns:Iterable[str],source_filenames:Iterable[str],dryrun=False)`
- **KEYS**
- `depends_from_result_dict(r: Dict[str,Any])->Iterable[int]`
- `append_depends_to_dict(to_submit:Dict[str,Any],previous_result: Dict[str,Any])`
- `parse_paths_from_dict(r: Doct[str,Any]) -> Doct[str,Any]`
- `submit_from_dict(r: Dct[str,Any]) -> Dict[str,Any]`
- **OpSubmitBroadcast(OperationOnSubdirectories,OperationOnFile)**
  - `__init__(self,filename,subdirectory_patterns:Iterable[str])`
  - `to_submit(self, r: RoutineOnDirectory) -> 'Observable[Dict[str,Any]]'`
  - `apply(self,r:RoutinOnDirectory) -> Dict[str,Iterable[Dict[str,str]]]`
  - `dryrun(self,r :RoutineOnDirectory) -> Dict[str,Iterable[Dict[str,str]]]`
- **OpSubmitSingleFile(OperationOnFile)**

- `__init__(self,filename:str)`
- `to_submit(self, r:RoutineOnDirectory)`
- `apply(self,r:RoutineOnDirectory) ->Doct[str,Any]`
- `dryrunn(self,r:RoutineOnDirectory) -> Dict[str,Any]`

## 5.2.8 scripts

*scripts*

### templates

- `make_mac.j2`
- `post_run.j2`
- `run_loacal.j2`
- `run.j2`
- `sbatch.j2`
- `shell.j2`

### `__init__.py`

#### `base.py`

- `ObjectWithTemplate(ObjectWithTemplateBase)`

#### `helper.py`

- `gate_simulation(mac_filename)`
- `gate_simulation_with_root_analysis(mac_filename,root_filename,analysis_c_filename)`      need to be corrected (gete->gate)

#### `shell.py`

- **`Script(ObjectWithTemplate)`**
  - `add_task(self,task)`
- **`Task`**
  - `render()`
- `TaskWithFileArg(Task)`
- `GateSimulation(TaskWithFileArg)`
- **`RootAnalysis(TaskWithFileArg)`**
  - `__init__(self, root_filename, c_file_name)`
  - `render(self)`

- **PygateAnalysis(TaskWithFileArg)**
  - `__init__(self, source=None, target=None, output=None, name=None, analysis_type=None)`
  - `render(self)`
- **Clean(Task)**
  - `render(self)`
- **Merge(TaskWithFileArg)**
- **ScriptRun(Script)**
  - `__init__(self, work_directory, tasks: Tuple[Task]=(), geant4_version='8.0', shell='bash', is_need_source_env=False, par`
  - `add_task(self, task)`
- **ScriptRunLocal(ScriptRun)**
  - `__init__(self, work_directory, tasks: Tuple[Task], geant4_version, shell='bash', is_nedd_source_env=False, local_work_d`
  - `add_task(self, task)`
- **ScriptPostRun(Script)**
  - `__init__(self, tasks: Tuple[Task]=(), geant4_version='8.0', shell='bash', is_need_source_env=False, partition='cpu')`
  - `add_task(self, task)`
- **ScriptMacTemplate(ObjectWithTemplate)**

## 5.2.9 test

test

## 5.2.10 utils

utils

`__init__.py`

`object_with_template.py`

- **EnvironmentOfPackage**
  - `__init__(self, pkg_name)`
  - `get_or_create_env(self)`
- `default_suffix(s: str, suffix: str, is_auto_add_point_prefix=True)`
- **ObjectWithTemplateBase**
  - `template_name(self)`
  - `render(self)`

### **strs.py**

### **syscall.py**

- `shell_call(commands: Iterable[str] or str, is_echo=True)`

### **typing.py**

- `JSONStr(str)`

## **5.2.11 \_\_init\_\_.py**

## **5.2.12 cleaner.py**

- **Cleaner**
  - `__init__(self,fs,config)`
  - `clean(self)`
  - `msg(self,path)`
  - `_clean_subs(self)`
  - `_clean_sources(self)`

## **5.2.13 conf.py**

- `KEYS`
- `SUBMIT_KEYS`
- `CLEAN_KEYS`
- `ANALYSIS_KEYS`
- **INIT\_KEYS**
  - `BROADCAST_KEYS`
  - `EXTERNAL_KEYS`
  - `MAC_KEYS`
  - `SHELL_KEYS`
- `PHANTOM_KEYS`

## **5.2.14 config\_maker.py**

- **ConfigMaker**
  - `_make_pygate_config(cls,fs,target,config_filename=None)`
  - `_make_mac_config(cls,fs,target,config=None,mac_config=None)`
  - `make(cls, fs,target,pygate_config=None,mac_config=None)`

### 5.2.15 config.py

### 5.2.16 configs.py

### 5.2.17 initializer.py

- **Initializer**

- `__init__(self,fs,config)`
- `_copy_sources_from_template(self)`
- `_make_mac(self)`
- `_make_phantom(self)`
- `_sub_dir_name(self,i)`
- `_nb_sub_dirs(self)`
- `_make_subdirs(self)`
- `_make_map_shell_scripts(self)`
- `_copy_sources_to_subdirs(self)`
- `pre_sub(self)`
- `make_sub(self)`

### 5.2.18 merger.py

- **Merger**

- `__init__(self,fs,config)`
- `_path_of_file_in_sub_dirs(self,base_filename)`
- `msg(self,method,target,sources)`
- `_hadd(self,task)`
- `_cat(self,task)`
- `_sum(self,task)`
- `_copy(self,task)`
- `merge(self)`

### 5.2.19 phantom.py

- **PhantomBinFileMaker:**

- `__init__(self,fs,phantom_files)`
- `_load_data(self,source)`
- `_make_bin(self,target,data)`
- `make(self)`

### 5.2.20 renderable.py

- **Renderable**
  - render(self) ->str

### 5.2.21 service.py

- make\_config(target='.')
- init(config,method)
- submit(config)
- merge(config)
- clean(config)

### 5.2.22 shell.py

- **ShellScriptBase**
  - \_\_init\_\_(self,fs,workdir:str, output:str,tasks:list,shell='zsh',version='7.2')
  - \_add\_if\_gate(self,task)
  - \_add\_if\_root(self,task)
  - \_get\_workdir\_on\_local\_and\_server(self)
  - \_make(self)
- **ShellScriptMerge(ShellScriptBase)**
  - \_\_init\_\_(self,fs,workdir:str,output:str,tasks:list,shell='zsh')
  - \_make(self)
- **ShellScriptMaker**
  - \_\_init\_\_(self,fs,config)
  - make(self)

### 5.2.23 submitter.py

- **Submitter**
  - \_\_init\_\_(self,fs,config)
  - \_get\_map\_info(self)
  - \_get\_merge\_info(self)
  - submit(self)
  - \_echo(self,info,fout)
  - \_slurm(self, run\_infos,post\_infos)
  - \_hqlf(self,run\_infos,post\_infos)

### 5.2.24 utils.py

- `get_scripts_path`
- `load_script(name)`
- `sub_dir_filters(config)`

## 5.3 Here is the command list of *pygate*

- **pygate**
  - **analysis**
    - \* `predefined`
    - \* `script`
  - `clean`
  - **generate**
    - \* `cfg`
    - \* **mac**
      - `predefined`
      - `script`
    - \* `mac_template`
    - \* `shell`
  - **init**
    - \* `auto`
    - \* `bcast`
    - \* `ext`
    - \* `subdir`
  - `merge`
  - `submit`

### 5.3.1 \$ pygate

Usage: `pygate [OPTIONS] COMMAND [ARGS]...`

**Options:**

- |                          |                                                 |
|--------------------------|-------------------------------------------------|
| <b>-c, --config TEXT</b> | config file name                                |
| <b>--no-config</b>       | ignore config file                              |
| <b>--dryrun</b>          | Do not do anything, just show expected results. |
| <b>--help</b>            | Show this message and exit.                     |

Commands:

```
analysis
clean
generate
init
merge
submit
```

### 5.3.2 \$ pygate analysis

Usage: pygate analysis [OPTIONS] COMMAND [ARGS]...

**Options:**

**--help** Show this message and exit.

Commands:

```
predefined
script
```

### 5.3.3 \$ pygate analysis predefined

Usage: pygate analysis predefined [OPTIONS]

**Options:**

**-n, --name TEXT** Predefined analysis type name.  
**-s, --source TEXT** Analysis source data filename.  
**-o, --output TEXT** Analysis target data filename.  
**--help** Show this message and exit.

### 5.3.4 \$ pygate analysis script

Usage: pygate analysis script [OPTIONS]

**Options:**

**-t, --target TEXT** Analysis .py filename.  
**-s, --source TEXT** Analysis source data filename.  
**-o, --output TEXT** Output filename.  
**--help** Show this message and exit.

### 5.3.5 \$ pygate clean

Usage: pygate clean [OPTIONS]

**Options:**

**-d, --subdirectories** remove subdirectories  
**-f, --root-files TEXT** remove files in work directory

**-s, --slurm-outputs** remove \*.out \*.err files  
**--help** Show this message and exit.

### 5.3.6 \$ pygate generate

Usage: pygate generate [OPTIONS] COMMAND [ARGS]...

**Options:**

**--help** Show this message and exit.

**Commands:**

|              |                                                    |
|--------------|----------------------------------------------------|
| cfg          | Generate initial config file.                      |
| mac          | Generate mac file.                                 |
| mac_template |                                                    |
| shell        | Generate shell script, pre run <b>or</b> post run. |

### 5.3.7 \$ pygate generate cfg

Usage: pygate generate cfg [OPTIONS]

Generate initial config file.

**Options:**

**-t, --target TEXT** Config file name.  
**-f, --format TEXT** Format of config file, json or yaml  
**--help** Show this message and exit.

### 5.3.8 \$ pygate generate mac

Usage: pygate generate mac [OPTIONS] COMMAND [ARGS]...

Generate mac file.

**Options:**

**--help** Show this message and exit.

**Commands:**

|            |                                          |
|------------|------------------------------------------|
| predefined | Generate mac file by predefined system.  |
| script     | Generate mac file by running a .py file. |

### 5.3.9 \$ pygate generate mac predefined

Usage: pygate generate mac predefined [OPTIONS]

Generate mac file by predefined system.

**Options:**

**-p, --predefined TEXT** Name of predefined system to generate mac file.  
**-c, --config TEXT** config filename to generate macs.

**-t, --target TEXT**    MAC filename, will passed to script or predefined method.  
**--help**                Show this message and exit.

### 5.3.10 \$ pygate generate mac script

Usage: pygate generate mac script [OPTIONS]

Generate mac file by running a .py file.

**Options:**

**-t, --target TEXT**    Filename of script to run to generate mac file.  
**-c, --config TEXT**    config filename to generate macs.  
**-o, --output TEXT**    MAC filename, will passed to script or predefined method.  
**--help**                Show this message and exit.

### 5.3.11 \$ pygate generate mac\_template

Usage: pygate generate mac\_template [OPTIONS]

**Options:**

**-f, --filename TEXT**    Show the file name.  
**--help**                Show this message and exit.

### 5.3.12 \$ pygate generate shell

Usage: pygate generate shell [OPTIONS]

Generate shell script, pre run or post run.

**Options:**

**--help**                Show this message and exit.

### 5.3.13 \$ pygate init

Usage: pygate init [OPTIONS] COMMAND [ARGS]...

**Options:**

**--help**                Show this message and exit.

Commands:

```
auto
bcast
ext      Copy external files.
subdir
```

### 5.3.14 \$ pygate init auto

Usage: pygate init auto [OPTIONS]

**Options:** `--mac-auto` `--mac-no-create` `--mac-force-create` `--help` Show this message and exit.

### 5.3.15 \$ pygate init bcast

Usage: pygate init bcast [OPTIONS]

**Options:**

`-t, --target INTEGER` Files to broadcast to subdirectories.  
`-e, --no-ext` Include all external files.  
`--help` Show this message and exit.

### 5.3.16 \$ pygate init ext

Usage: pygate init ext [OPTIONS]

Copy external files.

**Options:**

`--help` Show this message and exit.

### 5.3.17 \$ pygate init subdir

Usage: pygate init subdir [OPTIONS]

**Options:**

`-n, --nb-split INTEGER` Number of subdirectories.  
`-f, --sub-format TEXT` Subdirectories format str.  
`--help` Show this message and exit.

### 5.3.18 \$ pygate merge

Usage: pygate merge [OPTIONS]

**Options:**

`-t, --target TEXT` Target str.  
`-m, --method TEXT` Method str.  
`--help` Show this message and exit.

### 5.3.19 \$ pygate submit

Usage: pygate submit [OPTIONS]

**Options:**

- b, --broadcast TEXT** Broadcast file str.
- s, --single TEXT** Single str.
- help** Show this message and exit.

F



## CHAPTER 6

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`