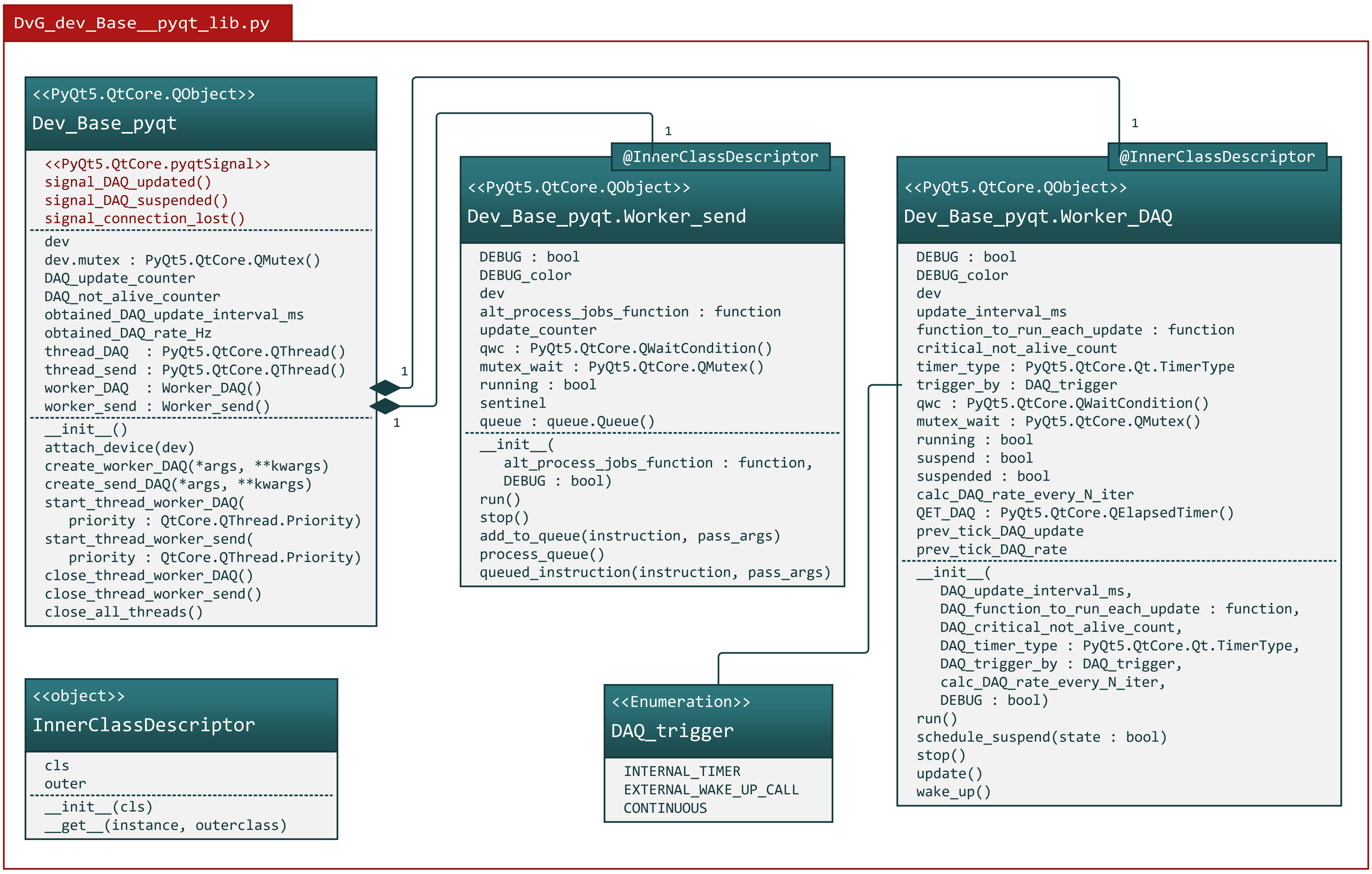

DvG_dev_Base__pyqt_lib.py

```
<<PyQt5.QtCore.QObject>>
Dev_Base_pyqt

<<PyQt5.QtCore.pyqtSignal>>
signal_DAQ_updated()
signal_DAQ_suspended()
signal_connection_lost()
------------------------------------
dev
dev.mutex : PyQt5.QtCore.QMutex()
DAQ_update_counter
DAQ_not_alive_counter
obtained_DAQ_update_interval_ms
obtained_DAQ_rate_Hz
thread_DAQ  : PyQt5.QtCore.QThread()
thread_send : PyQt5.QtCore.QThread()
worker_DAQ  : Worker_DAQ()
worker_send : Worker_send()
------------------------------------
__init__()
attach_device(dev)
create_worker_DAQ(*args, **kwargs)
create_send_DAQ(*args, **kwargs)
start_thread_worker_DAQ(
   priority : QtCore.QThread.Priority)
start_thread_worker_send(
   priority : QtCore.QThread.Priority)
close_thread_worker_DAQ()
close_thread_worker_send()
close_all_threads()
```

```
@InnerClassDescriptor
<<PyQt5.QtCore.QObject>>
Dev_Base_pyqt.Worker_send

DEBUG : bool
DEBUG_color
dev
alt_process_jobs_function : function
update_counter
qwc : PyQt5.QtCore.QWaitCondition()
mutex_wait : PyQt5.QtCore.QMutex()
running : bool
sentinel
queue : queue.Queue()
------------------------------------
__init__(
   alt_process_jobs_function : function,
   DEBUG : bool)
run()
stop()
add_to_queue(instruction, pass_args)
process_queue()
queued_instruction(instruction, pass_args)
```

```
@InnerClassDescriptor
<<PyQt5.QtCore.QObject>>
Dev_Base_pyqt.Worker_DAQ

DEBUG : bool
DEBUG_color
dev
update_interval_ms
function_to_run_each_update : function
critical_not_alive_count
timer_type : PyQt5.QtCore.Qt.TimerType
trigger_by : DAQ_trigger
qwc : PyQt5.QtCore.QWaitCondition()
mutex_wait : PyQt5.QtCore.QMutex()
running : bool
suspend : bool
suspended : bool
calc_DAQ_rate_every_N_iter
QET_DAQ : PyQt5.QtCore.QElapsedTimer()
prev_tick_DAQ_update
prev_tick_DAQ_rate
------------------------------------
__init__(
   DAQ_update_interval_ms,
   DAQ_function_to_run_each_update : function,
   DAQ_critical_not_alive_count,
   DAQ_timer_type : PyQt5.QtCore.Qt.TimerType,
   DAQ_trigger_by : DAQ_trigger,
   calc_DAQ_rate_every_N_iter,
   DEBUG : bool)
run()
schedule_suspend(state : bool)
stop()
update()
wake_up()
```

```
<<object>>
InnerClassDescriptor

cls
outer
------------------------------------
__init__(cls)
__get__(instance, outerclass)
```

```
<<Enumeration>>
DAQ_trigger

INTERNAL_TIMER
EXTERNAL_WAKE_UP_CALL
CONTINUOUS
```