Periodic Task Executor API

2016/03/04	0.1	Initial Version	Muhammad Z
2016/05/09	0.99	Final Draft	Muhammad Z

Summary

The Periodic Executor module should provide at minimum an implement for the following API.

API

You should provide support for the following API at least:

- Creation

```
PeriodicExecutor* PeriodicExecutor_Create(const char* _name,
                                          clockid_t _clk_id);
```

  Create a periodic executor that will measure time using given clock.
  Clockid_t specify the id of the system clock to be used. CLOCK_REALTIME, CLOCK_REALTIME_COARSE, CLOCK_MONOTONIC and similar provided it's supported by the underlying system.

- Cleanup

```
void PeriodicExecutor_Destroy(PeriodicExecutor* _executor);
```

  Destroy previously created executor. Cleaning all allocated memory and resources.

- Adding task to the executor

```
int PeriodicExecutor_Add(PeriodicExecutor* _executor,
                         Task _task,
                         void* _context,
                         size_t _period_ms);
```

_task is represented by a user provided function pointer.
_period_ms is the period of recurrence in milli seconds.

- Start running the executor

```
size_t PeriodicExecutor_Run(PeriodicExecutor* _executor);
```

Start running the tasks previously added to the executor or resume a previously paused Executor. This function will return in one of two cases:

1. The tasks were executed to completion.
2. The executor was paused

This function returns the number of tasks remaining inside the executor.

- Pause running the executor

```
size_t PeriodicExecutor_Pause(PeriodicExecutor* _executor);
```

Pause the execution of the executor. If a task is currently being executed then pause after it has been completed current cycle.
This function returns the number of tasks remaining inside the executor.