EMQF Python Sequence Manual

# Introduction

Eddie’s Message Queue Framework (EMQF) is a LabVIEW library for developing test applications. This framework provides a mechanism for Python sequences to be run and communicate with a top level application developed in LabVIEW.

The top level application will launch a python script that behaves as a Sequence Manager. This manager will connect to the top level app via TCP-IP.

The top level application will provide controls for the operator to select and run a python sequence file.

When the operator selects to launch the sequence, the top level application will send commands to the Sequence Manager to run the sequence.

The sequence file must have the following items:

* A class named ‘Seq’.
  + The SeqMgr object will be passed as an argument to this class.
  + The SeqMgr object contains methods to send status and data form the sequence to the top level app.
* That class will have a method named ‘main’.

The Sequence Manager will:

* Connect to the top level app to provide status and data.
* Catch and manage Python exceptions caused by the sequence file.
* Import the sequence file.
* Instantiate the Seq class
* Run the ‘main’ method.
* When the ‘main’ method completes, the sequence completed message will be sent to the top level app.

The sequence is responsible for:

* Periodically reporting status
  + A timeout is associated with the status messages.
  + If the timeout elapses between receiving status messages, the top level app will assume the sequence is in a bad state and it will be aborted (process killed).
* Monitoring flags for: Pause, Resume, and Abort.
  + These flags will be set/cleared by messages from the top level app.
* Catching and handling exceptions
  + The sequence should gracefully handle exceptions.
  + If it does not catch an exception, the sequence will crash causing an exception that is caught by Sequence Manager. This will cause the sequence to complete with an error result.

# Python Sequence Manager Deployment Options

## Local Python

This option is best for development stations where the sequences are frequently being updated.

Python is installed on the PC.

The emqf\_sequence\_manager.py is launched as a python script.

The list of available sequence files will be obtained from the .py files in a sequence folder provided by the top level app configuration.

## Bundled Exec

This option is best for production test stations where the sequences are infrequently changed.

Python does not need to be installed on the PC.

The emqf\_sequence\_manager is bundled to an exe file that also contains the sequence files.

The top level app will get a sequence file list buy launching emqf\_sequence\_manager.exe with an argument that will cause a sequence list to be provided on the standard output.

# Sequence Manager

The Sequence Manager is performed by the emqf\_sequence\_manager.py file. When launched this will instantiate the SeqMgr class then run the test\_exec method.

The application that launched the Sequence Manager will be referred to as the Master.

The test\_exec method will run a method in a separate thread that will manage communication with the master. It will connect to the master via TCP-IP client, and then it will respond to commands.

If it is unable to connect to the master, it will exit itself.

When the Sequence Manager exits, it will provide the following items on the standard output (STDOUT):

* Return code
* Space delimiter
* Description

If there is no error, the return code will be 0 and there will not be any description.

If there is an error, the return code will be a number, and the description will provide details.

## Command Line Arguments

Command line arguments will be processed.

Each argument is named:

<name>=<value>

If a value contains a string, then the argument must be put in in quotes.

All arguments are optional. If they are not provided, the default values will be used.

The following is a list of input arguments:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Value Desc** | **Default Value** | **Notes** |
| master\_ip | TCP-IP Address | localhost | Value stored |
| master-port | TCP-IP Port | 8500 | Value stored |

## TCP-IP Commands

The top-level application that launched the Test Manager will be referred to as Master.

The message protocol is:

<message length><command><tab character><arguments>

The message length is 4 bytes representing an unsigned 16 but integer. This is the length of the following message in bytes.

The rest of the message will be a string of ASCII characters.

If there are no arguments, there will be nothing beyond the command.

If there are arguments, a tab delimiter will be added and then will be followed by a JSON string that contains named arguments.

### From Master

#### List Sequences

Command: listSeq

Arguments: None

Action:

Return the message: Sequence List

#### Run Sequence

Command: seqRun

Arguments:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Value** | **Required** | **Description** |
| seq\_folder | Path to the sequence folder | No | If not provided, the default folder will be used. |
| seq\_name | Name of the sequence file | Yes |  |

Action:

Launch the sequence

Return the message: Status

#### Sequence Abort

Command: seqAbort

Arguments: none

Action:

Set the abort flag.

The sequence is responsible to check this flag and end itself.

Send status message

#### Sequence Terminate

Command: terminate

Arguments: none

Action:

Kill the sequence immediately.

Send status message

#### Pause

Command: seqPause

Arguments: none

Action:

Set the pause flag.

The sequence is responsible to check this flag and pause itself.

Send status message

#### Resume

Command: seqResume

Arguments: none

Action:

Clear the pause flag.

The sequence is responsible to check this flag and resume itself.

Send status message

#### Reset

Command: reset

Arguments: none

Action:

If a sequence is running, reply with NAK message.

Else:

Reset all status to default (no sequence running, no sequence results).

Send status message

#### Exit

Command: exit

Arguments: none

Action:

Send status message

Exit:

Kill any sequence that may be running

Close TCP-IP connection.

Exit execution.

### To Master

#### Sequence List

Command: seqList

Arguments:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Value** | **Required** | **Description** |
| sequences | array of sequence file names | Yes |  |

#### Status

Command: status

Arguments:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Value** | **Type** | **Description** |
| mgrState | Manager State | Int | See Sequence States |
| mgrDesc | Description | Str | String with description if error |
| seqName | Sequence Name | Str | Name of current or previous sequence |
| seqState |  | Int | State of current or previous sequence |
| seqResultCode |  | Int | State of current or previous sequence overall result |
| seqResultDesc |  | Str | Overall Result of current or previous sequence |
| seqRsltCnt |  | Str | Count of results |
| seqFailCnt |  | Str | Count of fail results |
| seqErrorCnt |  | Str | Count of error results |

Manager States:

|  |  |  |
| --- | --- | --- |
| **State** | **Description** | **Notes** |
| -1 | Error | An unexpected error occurred. See mgrDesc for details. |
| 0 | Idle | No sequence is running. |
| 1 | Running | A sequence is running. |

Sequence States:

|  |  |  |
| --- | --- | --- |
| **State** | **Description** | **Notes** |
| -1 | Error | There was an unhandled sequence exception, and the sequence was terminated. |
| 0 | None | No sequence has been run |
| 1 | Running | Running or resumed a sequence |
| 2 | Paused | The current sequence is Paused |
| 3 | Aborting | Abort Flag is set. Waiting for sequence to complete. |
| 4 | Completed | A sequence was run and it completed. |
| 5 | Aborted |  |

#### NAK

Description:

If the master sent an inappropriate command due to the sequence state, the sequence manager will respond with a NAK message.

Command: nak

Arguments:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Value** | **Type** | **Description** |
| nakCode | Code for error | Int |  |
| nakDescription | Description | Str | Description of error |

NAK Error States:

|  |  |  |
| --- | --- | --- |
| **State** | **Description** | **Notes** |
| 0 | Idle | No error. |
| -1 | Invalid Run command | The Seq Manager is in an error state. |
| -20 | Invalid Run command | A sequence is already running. |
| -21 | Invalid Run command | The sequence file path is invalid. |
| -3 | Invalid Reset command for current state | A sequence is already running. Can not reset. Use abort. |
| -4 | Invalid Pause command | A sequence is not running |
| -5 | Invalid Resume command | A sequence is not running |
| -6 | Invalid Abort command | A sequence is not running |
| -7 | Invalid Terminate command | A sequence is not running |