taran Documentation

Release 0.0.1

Jon Hadfield <jon@lessknown.co.uk>

August 14, 2016

Contents

< Description here >

Release v0.0.1.

CHAPTER 1

Installation

Using pip package manager:

```
$ pip install taran
```

From source:

```
$ git clone https://github.com/jonhadfield/taran
$ cd taran
$ python setup.py install
```

CHAPTER 2

API

2.1 taran.foreman

Foreman([configuration])	A template for all decision processors.

The Foreman class - An abstraction of the AWS SWF Decider operations

class taran.foreman.**Decision**(*name*, *type*, *schedule_to_start_timeout*, *start_to_close_timeout*, *schedule_to_close_timeout*, *task_list*, *input*)

__getnewargs__()
Return self as a plain tuple. Used by copy and pickle.

__getstate__()
Exclude the OrderedDict from pickling

__repr__()
Return a nicely formatted representation string

input
Alias for field number 6

name
Alias for field number 0

schedule_to_close_timeout
Alias for field number 4

schedule_to_start_timeout
Alias for field number 2

start_to_close_timeout
Alias for field number 3

task_list
Alias for field number 5

type
Alias for field number 1

class taran.foreman.**Foreman**(*configuration=None*)
A template for all decision processors.

configuration
module

The configuration a foreman needs in order to participate in the workflow.

get_activity_results(*activity=None*)
Get the result returned when the activity became completed.

get_workflow_history()
Get entire workflow history.

Returns a dict containing the entire workflow execution history

poll_for_decision_task()
Poll for an decision task from SWF and return if a task token has been provided.

Returns **task** – Details of the assigned task.

Return type dict

schedule_activity_tasks(*decisions=None*)
Retrieve the workflow history.

Args: decisions (List): A list of dictionaries containing details of the activities to schedule.

Parameters **decisions** (list) –

2.2 taran.starter

Starter([configuration])	Class that defines instances of starter that are used to perform checks and then execute a workflow.

This module provides a class that abstracts the configuration and the SWF 'start_workflow_execution' operation.

class taran.starter.**Starter**(*configuration=None*)
Class that defines instances of starter that are used to perform checks and then execute a workflow.

ensure_activity_type_exists(*activity_name=None*, *activity_version=None*, *activity_task_list=None*)
Check the activity type exists and create it if it doesn't.

Parameters

- **activity_name** (unicode) –
- **activity_version** (unicode) –
- **activity_task_list** (unicode) –

ensure_domain_exists(*domain_name=None*)
Return true if specified domain exists, otherwise create it.

ensure_workflow_type_exists(*workflow_name=None*, *workflow_version=None*)
Check the workflow type exists and create it if it doesn't.

Parameters

- **workflow_name** (unicode) –
- **workflow_version** (unicode) –

start_workflow()
Start the workflow.

2.3 taran.worker

Worker([configuration])	A template for all decision processors.

This module provides worker/actor specific methods to all child classes.

class taran.worker.**Worker**(*configuration=None*)
A template for all decision processors.

configuration
module

The configuration a worker needs in order to participate in the workflow.

activity_task_failed(*reason=None*, *details=None*)
Signal that activity task failed.

complete_activity_task(*result=u'Undefined'*)
Signal activity task as complete.

get_activity_results(*activity=None*)
Get a list of all results (when activity completed)

poll_for_activity_task()
Poll for an activity task from SWF and return if a task token has been provided.

Returns **task** – Details of the assigned task.

Return type dict

t

Symbols

A

C

D

E

F

G

I

N

P

S

T

W