Task Actions

Task actions provide features for running and interacting with asynchronous tasks managed by the XINA Run application or AWS Lambda platform.

Task Reference

Task Actions may reference existing tasks by either their numeric Task ID or a JSON object with the following format:

```
{
  "type": "ref",
  "ref": [<ref_id assigned when creating the task>]
}
```

The `ref_id` is an arbitrary, optional value defined in the Task Definition when a task is created. The `ref_id` does not have to be unique, implying the Task Action will be performed on all tasks that match the `ref_id`.

Task Actions

RUN

Run one or more asynchronous tasks.

Property	Value	Req	Default
action	`"run"`	?	

Example

```
{
  "action" : "run",
  "tasks"  : [ <task definition or reference>, ... ]
}
```

Task Definition

A task definition is used to define a new Task. It has a JSON object with the following format:

```
{
  "name"     : <string, task name>,
  "conf"     : <JSON object, format depends on task>,
  "parent"   : <long, parent task ID, optional>,
  "seq"      : <string, sequence name, optional>,
  "auto"     : <boolean, optional, default false>,
  "archive"  : <boolean, optional, default false - if true, task workspace will not be deleted>,
  "open"     : <boolean, optional, default false>,
  "desc"     : <string, optional>,
  "priority" : <int, optional>,
  "timeout"  : <int, ms, optional>,
  "ref_id"   : <long, optional - arbitrary value for Task Referencing>
}
```

CONCLUDE

Explicitly conclude a task. This is currently only used by AWS Lambda tasks, to notify the XINA server that the task has concluded. If a value is provided for `"delay"`, the server will wait that many milliseconds before concluding the task. This supports tasks that may have a longer cleanup or import period following the immediate task completion.

Property	Value	Req	Default
action	`"conclude"`	?	

Example

```
{
  "action" : "conclude",
  "task"   : <task ID>,
  "delay"  : <number, ms, 0-5000, optional, default 0>
}
```

CANCEL

Cancel one or more tasks.

Property	Value	Req	Default
action	`"cancel"`	?	

Only non-concluded tasks may be canceled. If "ignore" is false, and any specified tasks have concluded, an error will be thrown and no changes will occur. If "ignore" is true, all non-concluded specified tasks will be canceled,

and any concluded tasks will be ignored.

Example

```
{
  "action" : "cancel",
  "tasks"  : [ <task reference>, ... ],
  "ignore" : <boolean, optional, default false>
}
```

CLEAN

Permanently delete one or more asynchronous task records and any associated files.

Property	Value	Req	Default
action	`"clean"`	?	

Only concluded tasks may be cleaned. If `"ignore"` is `false`, and any specified tasks have not concluded, an error will be thrown and no changes will occur. If `"ignore"` is `true`, all concluded specified tasks will be cleaned, and any non-concluded tasks will be ignored.

Example

```
{
  "action" : "clean",
  "tasks"  : [ <task reference>, ... ],
  "ignore" : <boolean, optional, default false>
}
```

DESTROY

Cancel and clean one or more asynchronous tasks. Unlike the CANCEL and CLEAN actions, this will apply to all tasks regardless of the current task state.

Example

```
{
  "action" : "clean",
  "tasks"  : [ <task reference>, ... ]
}
```

Sequence Actions

PAUSE

Pause execution of one or more asynchronous task threads. This does not affect any task currently running in the thread, but future tasks assigned to the thread will not run until the thread is resumed.

Example

```
{
  "action"  : "pause",
  "threads" : [ <string, thread name>, ... ]
}
```

RESUME

Resume execution of one or more asynchronous task threads. If `"continue"` is `true` and a continuable task is currently locked on the thread, that task will be continued.

Example

```
{
  "action"   : "pause",
  "threads"  : [ <string, thread name>, ... ],
  "continue" : <boolean, optional, default false>

}
```

Revision #16
Created 12 January 2023 20:31:15 by Nick Dobson
Updated 15 January 2026 20:43:45 by Nick Dobson