DuraCloud REST API

DuraCloud REST API methods:

Notes

Each of the methods below has specific security requirements. See DuraCloud Security for more information
Examples calling the API defined below with the Unix utility "curl" can be found here

DuraStore

Purpose: DuraStore is the application through which DuraCloud manages storage. The DuraStore REST API provides access to storage by mediating the underlying storage provider APIs to allow access to multiple cloud storage options through a single API.

Store REST Methods

Get Stores

- Purpose: Provides a listing of available storage providers accounts (without credentials)
- Request: `GET https://host:port/durastore/stores`
- Parameters: None
- Response Code: 200 (on success)
- Response Body: XML similar to:

```
<storageProviderAccounts>
  <storageAcct isPrimary='1'>
    <id>1</id>
    <storageProviderType>AMAZON_S3</storageProviderType>
  </storageAcct>
  <storageAcct isPrimary="0">
    <id>2</id>
    <storageProviderType>RACKSPACE</storageProviderType>
  </storageAcct>
</storageProviderAccounts>
```

  - Note: The value of the isPrimary attribute is 1 for true and 0 for false

Space REST Methods

Get Spaces

- Purpose: Provides a listing of all of the spaces that a customer has created
- Request: `GET https://host:port/durastore/spaces ? (storeID)`
- Response Code: 200 (on success)
- Response Body: XML similar to:

```
<spaces>
  <space id="space1" />
  <space id="space2" />
</spaces>
```

Get Space

- Purpose: Provides a listing of the contents of a space along with space properties
- Request: `GET https://host:port/durastore/spaceID ? (storeID) (prefix) (maxResults) (marker)`
  - `storeID` (optional) - ID of the content storage provider to query (default is primary store)
  - `prefix` (optional) - Only retrieve content ids with this prefix (default is all content ids)
  - `maxResults` (optional) - The maximum number of content IDs to return in the list (default is 1000)
    note: the maximum allowable value for maxResults is 1000. Any larger value will be reduced to 1000.
  - `marker` (optional) - The content ID marking the last item in the previous set (default is the first set of ids)
- Response Code:
  - 200 (on success)
  - 404 (if the the given space does not exist)
- Response Body: XML similar to:

```
<space id="space1">
  <item>Image 1</item>
  <item>Image 2</item>
</space>
```

- Response Headers: All available space properties, example:

```
x-dura-meta-space-count: 65
x-dura-meta-space-created: 2016-04-14T01:40:47
```

Get Space Properties

- Purpose: Provides all space properties
- Request: `HEAD https://host:port/durastore/spaceID ? (storeID)`
- Response Code:
  - 200 (on success)
  - 404 (if the the given space does not exist)
- Response Headers: Same as for Get space (above)

Create Space

- Purpose: Creates a new space
- Request: `PUT https://host:port/durastore/spaceID ? (storeID)`
- Response Code:
  - 201 (on success)
  - 400 (if the content ID is invalid)
- Response Headers: Location of the new space (i.e. the URL used to create the space), example:

```
Location: https://myhost:8080/durastore/space1
```

Delete Space

- Purpose: Deletes a space
- Request: `DELETE` `https://host:port/durastore/spaceID ? (storeID)`
- Response Code:
  - 200 (on success)
  - 404 (if the the given space does not exist)
- Response Body: "Space $spaceID deleted successfully" (on success)

Get Space ACLs

- Purpose: Provides all space ACLs, with values of 'r' (read) and 'w' (read/write)
- Request: `HEAD` `https://host:port/durastore/acl/spaceID ? (storeID)`
- Response Code:
  - 200 (on success)
  - 404 (if the the given space does not exist)
- Response Headers: All available space ACLs, example:

```
x-dura-meta-acl-user0: WRITE
x-dura-meta-acl-user1: WRITE
x-dura-meta-acl-group-curators: READ
```

Set Space ACLs

- Purpose: Updates the ACLs associated with a space
- Request: `POST` `https://host:port/durastore/acl/spaceID ? (storeID)`
- Request Headers: For 'user' ACLs the header prefix must be 'x-dura-meta-acl-' and for 'groups' the header prefix must be 'x-dura-meta-acl-group-'. Allowable values for ACL headers are: 'READ' and 'WRITE'.
  Example:

```
x-dura-meta-acl-user0: WRITE
x-dura-meta-acl-user1: WRITE
x-dura-meta-acl-group-curators: READ
```

- Response Code:
  - 200 (on success)
  - 404 (if the the given space does not exist)
- Response Body: "Space $spaceID ACLs updated successfully" (on success)

Content REST Methods

Get Content

- Purpose: Retrieves a piece of content along with its properties
- Request: `GET` `https://host:port/durastore/spaceID/contentID ? (storeID) (attachment)`
  - if attachment param value is true, a Content-Disposition header is included with the response
- Response Code: 200 (on success)
- Response Body: The content stream
- Response Headers: All available content properties, example:

```
Content-Type: text/plain
Content-Length: 5732
Content-MD5: 3456709234785097473839202
ETag: 3456709234785097473839202
x-dura-meta-content-name: Testing Content
x-dura-meta-content-owner: JSmith
```

Get Content Properties

- Purpose: Retrieves the properties of a piece of content without the content itself
- Request: `HEAD` `https://host:port/durastore/spaceID/contentID ? (storeID)`
- Response Code: 200 (on success)
- Response Headers: Same as Get content (above)

Store Content

- Purpose: Adds a piece of content to the store
- Request: `PUT` `https://host:port/durastore/spaceID/contentID ? (storeID)`
- Request Body: Content to be added

- Request Headers: Properties about the content, example:

```
Content-Type: text/plain
Content-MD5: 4cd56e137a93a1accb43c5d32f4afffb
x-dura-meta-content-name: Testing Content
x-dura-meta-content-owner: JSmith
```

- Response Code:
  - 201 (on success)
  - 400 (if the content ID is invalid)
  - 404 (if the the given space does not exist)
  - 409 (if the provided checksum did not match the stored content checksum)
  - 500 (on error)
- Response Headers:
  - MD5 checksum of stored content
  - ETag of stored content
  - Location of the new content (i.e. the URL used to create the content), example:

```
Content-MD5: 4cd56e137a93a1accb43c5d32f4afffb
ETag: 4cd56e137a93a1accb43c5d32f4afffb
Location: https://myhost:8080/durastore/space1/content1
```

- Usage Notes
  - When the optional Content-MD5 header is included, the final checksum of the stored file is compared against the MD5 value included in the header to ensure that the file was stored correctly. If the header is not included, an MD5 checksum is computed as the file is transferred to storage, and that value is used in the final comparison.
  - All properties to be set must be included as a request header with the prefix "x-dura-meta-". Any header using the "x-dura-meta-" prefix will be stored as a content property, with a few exceptions, which are used for specific other purposes:
    - Headers used as part of the Copy Content request

```
x-dura-meta-copy-source
x-dura-meta-copy-source-store
```

    - Headers used to provide details about a space

```
x-dura-meta-space-count
x-dura-meta-space-created
```

    - Headers used as part of the Set Space ACLs call (the * is replaced by the user or group name)

```
x-dura-meta-acl-*
x-dura-meta-acl-group-*
```

    - Headers used internal to DuraCloud

```
x-dura-meta-content-mimetype (set Content-Type header instead)
x-dura-meta-content-size (automatically written as Content-Length header)
x-dura-meta-content-checksum (automatically written as Content-MD5 header)
x-dura-meta-content-modified (automatically written as Last-Modified header)
```

    - Headers used by the DuraCloud SyncTool to automatically capture file details

```
x-dura-meta-creator
x-dura-meta-content-file-created
x-dura-meta-content-file-modified
x-dura-meta-content-file-last-accessed
x-dura-meta-content-file-path
```

  - Use only US-ASCII characters for property names and values
  - There is a 2 KB total size limit on all content properties (this includes both auto-generated and user contributed properties.)
  - The "x-dura-meta-" prefix is case-sensitive (make sure your clients do not automatically change case.)

Copy Content

- Purpose: Copies a piece of content from a source space to a destination space within a given store
- Request: `PUT https://host:port/durastore/spaceID/contentID ? (storeID)`
- Request Body: must not exist
- Request Headers: Copy source, example:

```
x-dura-meta-copy-source: space-id/content-id
```

- Optional Request Headers: Copy source store, example:

```
x-dura-meta-copy-source-store: storeId
```

- Response Code: 201 (on success)
- Response Headers:
  - MD5 checksum of stored content
  - ETag of stored content
  - Location of the new content (i.e. the URL used to create the content), example:

```
Content-MD5: 4cd56e137a93a1accb43c5d32f4afffb
ETag: 4cd56e137a93a1accb43c5d32f4afffb
Location: https://myhost:8080/durastore/space1/content1
```

- Usage Notes
  - The properties associated with the source content item are copied to the destination content item.
  - The source and destination spaces may be the same.
  - Including the optional header indicates that the copy action should retrieve the source file from a space in the specified storage provider. This allows for copying a file from one storage provider to another.

Set Content Properties

- Purpose: Updates the properties associated with a piece of content. **Note:** You must include ALL properties you would like associated with the given content item in this call. Any properties that exist before this call but are not included in the call itself will be removed. This is to allow for both adding and removing properties.
- Request: `POST https://host:port/durastore/spaceID/contentID ? (storeID)`
- Request Headers: Same as Store content (above)
- Response Code: 200 (on success)
- Response Body: "Content $contentID updated successfully"

Delete Content

- Purpose: Removes a piece of content from the store
- Request: `DELETE https://host:port/durastore/spaceID/contentID ? (storeID)`
- Response Code: 200 (on success)
- Response Body: "Content $contentID deleted successfully"

Audit Log REST Methods

Get Audit Log

- Purpose: Returns the latest audit for a given store and space
- Request: `GET https://host:port/durastore/audit/{spaceId} ? (storeID)`
  - `spaceID - ID of the space for which the audit log will be retrieved`
  - `storeID (optional) - ID of the content storage provider to query (default is primary store)`
- Response Code: 200 (on success), 404 if audit logs were not found.
- Response Body: TSV in chronological order with the following fields.

```
ACCOUNT        STORE_ID        SPACE_ID        CONTENT_ID        CONTENT_MD5
CONTENT_SIZE        CONTENT_MIMETYPE        CONTENT_PROPERTIES        SPACE_ACLS
SOURCE_SPACE_ID        SOURCE_CONTENT_ID        TIMESTAMP        ACTION        USERNAME
mysubdomain        51        myspace        image-01.jpg
b1978f9fc4fe9448e05b83bbe6b98109        81214        image/jpeg        {"content-mimetype" : "image
/jpeg"}                {}                2014-09-10T15:54:42.042        ADD_CONTENT        root
```

Manifest REST Methods

Get Manifest

- Purpose: Returns the manifest for a given space and storeId
- Request: `GET https://host:port/durastore/manifest/{spaceId} ? (storeID) (format)`
  - `spaceID - ID of the space for which the manifest will be retrieved`
  - `storeID (optional) - ID of the content storage provider to query (default is primary store)`
  - `format (optional) - TSV or BAGIT (default is TSV)`
- Response Code: 200 (on success), 404 if manifest was not found.
- Response Body: TSV in chronological order with the following fields.

TSV results

```
space-id        content-id        MD5
auditlogs        localhost/51/auditlogs/localhost_51_auditlogs-2014-09-10-15-56-07.tsv
6992f8e57dafb17335f766aa2acf5942
auditlogs        localhost/51/photos/localhost_51_photos-2014-09-10-15-55-01.tsv
820e786633fb495db447dc5d5cf0b2bd
```

Generate Manifest

- Purpose: Asynchronously generates a gzipped manifest for a given space and storeId. This approach may be preferable if you wish to obtain a manifest for a larger space. We recommend considering this option for spaces that are larger than 100K items.
- Request: `POST https://host:port/durastore/manifest/{spaceId} ? (storeID) (format)`
  - `spaceID - ID of the space for which the manifest will be generated`
  - `storeID (optional) - ID of the content storage provider to query (default is primary store)`
  - `format (optional) - TSV or BAGIT (default is TSV)`
- Response Code: 202 (on success), 404 if manifest was not found.
- Response Body: We are processing your manifest generation request. To retrieve your file, please poll the URI in the Location header of this response
- Response Headers:

TSV results

```
Location: <URI-of-generated-manifest>
```

Storage Report REST Methods

Get Storage Reports by Space

- Purpose: Returns storage report summaries for a space. Report values are averaged based on the grouping internal (if groupBy=month, all data points within each month are averaged to provide an aggregate result).
- Request: `GET https://host:port/durastore/report/space/{spaceId} ? (storeID) (start) (end) (groupBy)`
  - `spaceID - ID of the space for which the storage report will be retrieved`
  - `storeID (optional) - ID of the content storage provider to query (default is primary store)`
  - `start (optional) - Timestamp in epoch milliseconds which defines the starting point for results. Any data points which are prior to this value are not included.`
  - `end (optional) - Timestamp in epoch milliseconds which defines the end point for results. Any data points which are after this value are not included.`
    - `usage note: To ensure that all expected data points are included, set the end timestamp to the very end of the final interval (e.g. 23:59:59 on the last day of the week/month)`
  - `groupBy (optional) - Grouping interval which allows for averaged results for days, weeks, and months. Valid values are: "day", "week", and "month" (default is day)`
- Response Code: 200 (on success)
- Response Body: JSON array of storage report details

JSON results


```
[
  {"timestamp":1312588800000,"accountId":"<account-id>","spaceId":"<space-id>","storeId":"<store-id>","
byteCount":1000,"objectCount":10},
  {"timestamp":1315008000000,"accountId":"<account-id>","spaceId":"<space-id>","storeId":"<store-id>","
byteCount":1000,"objectCount":10},
  {"timestamp":1315526400000,"accountId":"<account-id>","spaceId":"<space-id>","storeId":"<store-id>","
byteCount":1000,"objectCount":10}
]
```


Get Storage Reports by Store

- Purpose: Returns storage report summaries for all content in a storage provider. Report values are averaged based on the grouping internal (if groupBy=month, all data points within each month are averaged to provide an aggregate result).
- Request: `GET https://host:port/durastore/report/store ? (storeID) (start) (end) (groupBy)`
  - `storeID (optional) - ID of the content storage provider to query (default is primary store)`
  - `start (optional) - Timestamp in epoch milliseconds which defines the starting point for results. Any data points which are prior to this value are not included.`
  - `end (optional) - Timestamp in epoch milliseconds which defines the end point for results. Any data points which are after this value are not included.`
    - `usage note: To ensure that all expected data points are included, set the end timestamp to the very end of the final interval (e.g. 23:59:59 on the last day of the week/month)`
  - `groupBy (optional) - Grouping interval which allows for averaged results for days, weeks, and months. Valid values are: "day", "week", and "month" (default is day)`
- Response Code: 200 (on success)
- Response Body: JSON array of storage report details

JSON results


```
[
  {"timestamp":1312588800000,"accountId":"<account-id>","storeId":"<store-id>","byteCount":1000,"
objectCount":10},
  {"timestamp":1315008000000,"accountId":"<account-id>","storeId":"<store-id>","byteCount":1000,"
objectCount":10},
  {"timestamp":1315526400000,"accountId":"<account-id>","storeId":"<store-id>","byteCount":1000,"
objectCount":10}
]
```


Get Storage Reports for all Spaces in a Store (in a single day)

- Purpose: Returns storage report summaries for all spaces in a storage provider on a single day.
- Request: `GET https://host:port/durastore/report/store/{date} ? (storeID)`
  - `date - Timestamp in epoch milliseconds which specifies the requested day`
  - `storeID (optional) - ID of the content storage provider to query (default is primary store)`
- Response Code: 200 (on success)
- Response Body: JSON array of storage report details

JSON results

```
[
  {"timestamp":1312588800000,"accountId":"<account-id>","spaceId":"<space-id-1>","storeId":"<store-id>","byteCount":1000,"objectCount":10},
  {"timestamp":1315008000000,"accountId":"<account-id>","spaceId":"<space-id-2>","storeId":"<store-id>","byteCount":1000,"objectCount":10},
  {"timestamp":1315526400000,"accountId":"<account-id>","spaceId":"<space-id-3>","storeId":"<store-id>","byteCount":1000,"objectCount":10}
]
```

Bit Integrity REST Methods

Get Bit Integrity Report

- Purpose: Retrieves the latest bit integrity report for a given space and store
- Request: `GET https://host:port/durastore/bit-integrity/{spaceId} ? (storeID)`
  - `Optional parameter 'storeID': if not set, primary storage provider is used.`
- Response Code: 200 (on success), 404 if space doesn't exist, 204 if no report is available for that space.
- Response Headers:
  - Bit-Integrity-Report-Completion-Date: yyyy-MM-ddTHH:mm:ss
  - Bit-Integrity-Report-Result: (SUCCESS or FAILURE)
- Response Body: TSV with the following fields.

TSV results

```
date-checked        account        store-id        store-type        space-id        content-id
result        content-checksum        provider-checksum        manifest-checksum        details
```

Get Bit Integrity Report Properties

- Purpose: Retrieves details about the latest bit integrity report for a given space and store, but not the report itself
- Request: `HEAD https://host:port/durastore/bit-integrity/{spaceId} ? (storeID)`
  - `Optional parameter 'storeID': if not set, primary storage provider is used.`
- Response Code: 200 (on success), 404 if space doesn't exist, 204 if no report is available for that space.
- Response Headers: same as for Get Bit Integrity Report (above)

Task REST Methods

Tasks are used to perform storage provider actions which cannot be performed in a generic manner across multiple providers.

Get Tasks

- Purpose: Provides a listing of all of the supported tasks for a given provider. Note that if no storeID parameter is included, the task listing is provided for the primary storage provider.
- Request: `GET https://host:port/durastore/task ? (storeID)`
- Response Code: 200 (on success)
- Response Body: XML similar to:

```
<list>
  <string>task1</string>
  <string>task2</string>
</list>
```

Perform Task

- Purpose: Performs a particular task. Note that most tasks can be performed by only one storage provider type.
- Request: `POST https://host:port/durastore/task/taskName ? (storeID)`
- Request Body: Parameters for task. Each task will expect parameters in a specific format, see task listing for more details.
- Response Code: 200 (on success)
- Response Body: Response value for task, format varies by task.

Tasks

Amazon S3 Storage Provider

taskName	Name	Description	Request Body	Response Body
enable-streaming	Enable Streaming task	Enables RTMP streaming for all files within a DuraCloud space through the use of Amazon's Cloudfront streaming capability. This task may take up to 15 minutes to complete. When this call completes, two new properties will have been added to the set of properties for the specified space: • streaming-host - this is the RTMP host value, which can be used to generate URLs for open streams • streaming-type - will either be OPEN or SECURE, depending on the value of the secure parameter provided when streaming was enabled	`{ "spaceId" : "", "secure" : "" }` spaceId - Name of the space for which streaming is to be enabled secure - true or false, should streaming be secured	`{ "result" : "", "streamingHost" : "" }` result - Text indicating the results of the task streamingHost - the host name of the streaming endpoint
disable-streaming	Disable Streaming task	Disables streaming by removing the ability for Cloudfront to access files within a space. This does not remove the streaming distribution, only disables its use, so enabling streaming on the same space again can be performed much more quickly. Some content in the space may continue to be available for streaming up to 24 hours after streaming has been disabled.	`{ "spaceId" : "" }` spaceId - Name of the space for which streaming is to be disabled	`{ "result" : "" }` result - Text indicating the results of the task
delete-streaming	Delete Streaming task	Removes a streaming distribution created by the enable-streaming task. This task should be performed after performing the disable-streaming task. This task may take up to 15 minutes to complete, after which no content in the space will be available for streaming.	`{ "spaceId" : "" }` spaceId - Name of the space for which streaming is to be deleted	`{ "result" : "" }` result - Text indicating the results of the task

get-url	Get URL task	Retrieves a URL for a media file that is streamed through Cloudfront via an open distribution	`{ "spaceId" : "", "contentId" : "", "resourcePrefix" : "" }` spaceId - Name of the space in which the streamed content is stored contentId - Name of the content item to be streamed resourcePrefix - A prefix on the content item which may be required by the streaming viewer. (e.g. an mp4 file may need a prefix of "mp4:") (optional)	`{ "streamUrl" : "" }` streamUrl - The URL to be used for streaming the requested content
get-signed-url	Get Signed URL task	Retrieves a signed URL for a media file that is streamed through Cloudfront via a secure distribution	`{ "spaceId" : "", "contentId" : "", "minutesToExpire" : "", "ipAddress" : "" "resourcePrefix" : "" }` spaceId - Name of the space in which the streamed content is stored contentId - Name of the content item to be streamed minutesToExpire - Number of minutes until the generated URL expires and the stream can no longer be played (optional, default is 480) ipAddress - IP address range where requests to stream must originate, in CIDR notation (e.g. 1.2.3.4/32) (optional) resourcePrefix - A prefix on the content item which may be required by the streaming viewer. (e.g. an mp4 file may need a prefix of "mp4:") (optional)	`{ "signedUrl" : "" }` signedUrl - The URL to be used for streaming the requested content
set-storage-policy	Set Storage Policy	Sets the S3 bucket lifecycle policies associated with a given space. This task is restricted to DuraCloud service administrators.	`{ "spaceId" : "", "storageClass" : "", "daysToTransition" : 0 }` spaceId - Name of the space for which the storage policy should be set storageClass - One of "STANDARD_IA", "REDUCED_REDUNDANCY", or "GLACIER" daysToTransition - Number of days content should remain at standard storage before being transitioned to the new storage class	`{ "result" : "" }` result - Text indicating the results of the task
noop	Test task	Provides a simple way to test the calling of tasks	None	"Success"

Amazon Glacier Storage Provider

taskName	Name	Description	Request Body	Response Body
restore-content	Restore Content task	Provides the capability to request that specific content items stored in Glacier be retrieved. Content items which are retrieved are made available 3-5 hours after this request is made, and remains available for 2 weeks.	Name of the space and the content item in the form: spaceID /contentID	Text indicating that a restore action has been initiated (or that a restore is already in progress, in the case of duplicate requests.)

Snapshot Storage Provider

taskName	Name	Description	Request Body	Response Body
create-snapshot	Create Snapshot task	Creates a snapshot by collecting details of the snapshot and passing the request down to a bridge application which makes a copy of the contents of the space.	`{ "spaceId" : "", "description" : "", "userEmail" : "" }`	`{ "snapshotId" : "", "status" : "" }`

get-snapshot	Get Snapshot task	Retrieves the status and details of a snapshot action	{
			
			"snapshotId" : ""
			}
				{
				
				"snapshotId" : "",
				
				"snapshotDate" : "",
				
				"status" : "",
				
				"sourceHost" : "",
				
				"sourceSpaceId" : "",
				
				"sourceStoreId" : "",
				
				"description" : "",
				
				"contentItemCount" : "",
				
				"totalSizeInBytes" : ""
				
				"alternateIds" : ["", ""]
				}

cleanup-snapshot	Clean Up Snapshot task	Handles the removal of content items in a space after a snapshot has taken place	{
			
			"spaceId" : ""
			}
				{
				
				"contentExpirationDays" : ""
				}

complete-snapshot	Complete Snapshot task	Completes the snapshot process	{
			
			"spaceId" : ""
			}
				{
				
				"result" : ""
				}

complete-cancel-snapshot	Complete the cancellation of a snapshot	Handles the removal of any space properties, .collection-snapshot.properties file, and snapshot related user permissions. It should be called by the bridge after it has finished its cancellation process.	`{ "spaceId" : "" }`	`{ "result" : "text description of result" }`
restart-snapshot	Restart Snapshot task	Restarts the snapshot process if a failure occurred while transferring from DuraCloud to the bridge.	`{ "snapshotId" : "" }`	`{ "snapshotId" : "", "status" : "" }`
get-snapshots	Get List of Snapshots task	Retrieves a listing of all snapshots which have been created	None	`{ "snapshots" : [ { "snapshotId" : "", "description" : "", "status" : "" }, ..., ... ] }`
get-snapshot-contents	Get List of Snapshot Contents task	Retrieves a listing of the contents of a particular snapshot	`{ "snapshotId" : "", "pageNumber" : 0, "pageSize" : 1000, "prefix" : "" }`	`{ "totalCount" : 0, "contentItems" : [{ "contentId" : "", "contentProperties" : { "" : "" } }] }`

get-snapshot-history	Get Snapshot History task	Retrieves a listing of events which have occurred in the history of a particular snapshot	{
			"snapshotId" : "",
			"pageNumber" : 0,
			"pageSize" : 0
			}
				{
				"totalCount" : 0,
				"historyItems" :
				[{
				"history" : "",
				"historyDate" : 0
				}]
				}

request-restore-snapshot	Request a snapshot restore	Sends a restore request to an duracloud admin level user. This call can be made by user with access to the snapshot in question. Action on the part of the admin receiving the request is required to initiate a restore. The value of the user email address parameter will be used for notification purposes once the restore begins.	{
			"snapshotId" : "",
			"userEmail" : ""
			}
				{
				"description" : ""
				}

restore-snapshot	Restore Snapshot task	Initiates the restoration of a snapshot to a DuraCloud space. This call requires admin access.	{
			"snapshotId" : "",
			"userEmail" : ""
			}
				{
				"spaceId" : "",
				"restoreId" : "",
				"status" : ""
				}

complete-restore	Complete Restore task	Completes the restoration action by setting up an expiration policy for restored content	{
			"spaceId" : "",
			"daysToExpire" : 1
			}
				{
				"result" : ""
				}

get-restore	Get Snapshot Restore task	Retrieves the status and details of a restore action. Note that you must specify either the snapshotId or the restoreId, but not both. Specifying the snapshotId will return the most recent restoration matching that snapshotId. Specifying the restoreId you will get back the restoration matching that ID (as you would expect).	`{ "snapshotId" : "", "restoreId" : "" }`	`{ "restoreId" : "", "snapshotId" : "", "status" : "", "startDate" : "", "endDate" : "", "statusText" : "", "destinationHost" : "", "destinationPort" : "", "destinationStoreId" : "", "destinationSpaceId" : "" }`