



管理工作流 Astra Automation

NetApp
August 11, 2025

This PDF was generated from https://docs.netapp.com/zh-cn/astra-automation/workflows/workflows_before.html on August 11, 2025. Always check docs.netapp.com for the latest.

目录

- 管理工作流 1
 - 准备使用管理工作流 1
 - 一般准备 1
 - 工作流类别 1
 - 其他注意事项 1
- 应用程序控制 2
 - 列出应用程序 2
 - 获取应用程序 3
 - 管理应用程序 4
 - 取消管理应用程序 5
- 应用程序保护 6
 - 列出快照 6
 - 列出备份 8
 - 为应用程序创建快照 9
 - 为应用程序创建备份 10
 - 删除快照 11
 - 删除备份 12
 - 创建数据保护策略 13
- 克隆和还原应用程序 14
 - 克隆应用程序 14
 - 从快照克隆应用程序 16
 - 从备份克隆应用程序 17
 - 从备份还原应用程序 18
- 命名空间 19
 - 列出命名空间 19
- 支持 21
 - 列出通知 21
 - 删除失败的应用程序 23

管理工作流

准备使用管理工作流

您可以在管理Astra受管集群中的应用程序时使用这些工作流。



NetApp 可以随时扩展和改进这些工作流，因此您应定期查看这些工作流。

一般准备

在使用任何 Astra 工作流之前，请务必查看 ["准备使用这些工作流"](#)。

工作流类别

管理工作流按不同类别进行组织，以便更容易找到所需的工作流。

类别	Description
应用程序控制	通过这些工作流，您可以控制受管应用程序和非受管应用程序。您可以列出应用程序以及创建和删除受管应用程序。
应用程序保护	您可以使用这些工作流通过快照和备份保护受管应用程序。
克隆和还原应用程序	这些工作流介绍了如何克隆和还原受管应用程序。
支持	有多个工作流可用于调试和支持您的应用程序以及常规 Kubernetes 环境。

其他注意事项

使用管理工作流时，还需要注意一些其他注意事项。

克隆应用程序

克隆应用程序时，需要考虑一些事项。以下所述参数是 JSON 输入的一部分。

源集群标识符

值 `sClusterID` 始终表示安装原始应用程序的集群。

集群标识符

值 `clusterID` 用于标识要安装新应用程序的集群。

- 在同一集群中克隆时，`clusterID` 和 `sClusterID` 具有相同的值。
- 在集群间克隆时，这两个值不同，并且 `clusterID` 应为目标集群的 ID。

命名空间

`namespace` 值必须与原始源应用程序不同。此外，克隆的命名空间不存在，Astra 将创建它。

备份和快照

您可以选择使用 `backupID` 或 `snapshotID` 参数从现有备份或快照克隆应用程序。如果不提供备份或快照，则 Astra 将首先创建应用程序的备份，然后从备份中克隆。

还原应用程序

以下是还原应用程序时需要考虑的几个事项。

- 还原应用程序与克隆操作非常相似。
- 还原应用程序时，您必须提供备份或快照。

应用程序控制

列出应用程序

您可以列出当前由 Astra 管理的应用程序。您可以在查找特定应用程序的快照或备份时执行此操作。

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
获取	/account/ {account_id} /k8s/v2/apps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
包括	查询	否	也可以选择要在响应中返回的值。

curl 示例：返回所有应用程序的所有数据

```
curl --request GET \  
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

curl 示例：返回所有应用程序的名称，ID 和状态

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps?include=name,id,
state" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 输出示例

```
{
  "items": [
    [
      "mysql",
      "4ee2b8fa-3696-4f32-8879-399792f477c3",
      "ready"
    ],
    [
      "postgresql",
      "3b984474-e5c9-4b64-97ee-cdeb9bcd212e",
      "ready"
    ],
  ],
  "metadata": {}
}
```

获取应用程序

您可以检索描述单个应用程序的所有资源变量。

开始之前

您必须具有要检索的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。

第1步：获取应用程序

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
获取	/accounts/ {account_id} /k8s/v2/apps/ {app_id}

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	要检索的应用程序的ID值。

curl 示例：返回应用程序的所有数据

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps/$APP_ID" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

管理应用程序

您可以基于Astra已知的应用程序在特定命名空间中创建托管应用程序。在将应用程序管理或定义到Astra后、您可以通过创建备份和快照来对其进行保护。

第1步：选择命名空间

执行工作流 ["列出命名空间"](#) 并选择命名空间。

第2步：选择集群

执行工作流 ["列出集群"](#) 并选择集群。

第3步：管理应用程序

执行以下REST API调用以管理此应用程序。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v2/apps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
JSON	body	是的。	提供确定要管理的应用程序所需的参数。请参见以下示例。

curl 示例：管理应用程序

```
curl --request POST \
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps" \
--include \
--header "Content-Type: application/astra-app+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data @JSONinput
```

JSON 输入示例

```
{
  "clusterID": "7ce83fba-6aa1-4e0c-a194-26e714f5eb46",
  "name": "subtext",
  "namespaceScopedResources": [{"namespace": "kube-matrix"}],
  "type": "application/astra-app",
  "version": "2.0"
}
```

取消管理应用程序

您可以删除不再需要的受管应用程序。删除受管应用程序也会删除关联的计划。

开始之前

您必须具有要取消管理的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。
删除应用程序时，不会自动删除其备份和快照。如果您不再需要备份和快照，应在删除应用程序之前将其删除。

取消管理应用程序

执行以下REST API调用以删除此应用程序。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
删除	/accounts/ {account_id} /k8s/v2/apps/ {app_id}

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识要删除的应用程序。

curl 示例：删除受管应用程序

```
curl --request DELETE \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps/$APP_ID" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

应用程序保护

列出快照

您可以列出为特定应用程序创建的快照。

开始之前

您必须具有要列出其快照的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。

列出快照

执行以下REST API调用以列出快照。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
获取	/accouns/ {account_id} /k8s/v1/apps/ {app_id} /appSaps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识拥有列出快照的应用程序。
count	查询	否	如果为 count=true ，则快照数包含在响应的元数据部分中。

curl 示例：返回应用程序的所有快照

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appSnaps" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

curl 示例：返回应用程序和计数的所有快照

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appSnaps?count=true" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

JSON 输出示例

```
{  
  "items": [  
    {  
      "type": "application/astra-appSnap",  
      "version": "1.1",  
      "id": "1ce34da4-bb0a-4926-b925-4a5d85dda8c2",  
      "hookState": "success",  
      "metadata": {  
        "createdBy": "a530e865-23e8-4e2e-8020-e92c419a3867",  
        "creationTimestamp": "2022-10-30T22:44:20Z",  
        "modificationTimestamp": "2022-10-30T22:44:20Z",  
        "labels": []  
      },  
      "snapshotAppAsset": "0ebfe3f8-40ed-4bdc-88c4-2144fbda85a0",  
      "snapshotCreationTimestamp": "2022-10-30T22:44:33Z",  
      "name": "snapshot-david-1",  
      "state": "completed",  
      "stateUnready": []  
    },  
  ],  
  "metadata": {}  
}
```

列出备份

您可以列出为特定应用程序创建的备份。

开始之前

您必须具有要列出备份的应用程序的ID。如果需要，您可以使用此[工作流 "列出应用程序"](#) 以查找应用程序。

列出备份

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
获取	/accounts/ {account_id} /k8s/v1/apps/ {app_id} /appBackups

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识拥有列出备份的受管应用程序。

curl 示例：返回应用程序的所有备份

```
curl --request POST \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appBacku  
ps" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

JSON 输出示例

```
{
  "items": [
    {
      "type": "application/astra-appBackup",
      "version": "1.1",
      "id": "8edeb4a4-fd8b-4222-a559-1013145b28fc",
      "name": "backup-david-oct28-1",
      "bucketID": "a443e58f-59bd-4d45-835a-1bc7813f659a",
      "snapshotID": "dfe237cb-57b7-4576-af4d-00ba3a8f2828",
      "state": "completed",
      "stateUnready": [],
      "hookState": "success",
      "totalBytes": 205219132,
      "bytesDone": 205219132,
      "percentDone": 100,
      "metadata": {
        "labels": [
          {
            "name": "astra.netapp.io/labels/read-only/triggerType",
            "value": "backup"
          }
        ],
        "creationTimestamp": "2022-10-28T21:58:37Z",
        "modificationTimestamp": "2022-10-28T21:58:55Z",
        "createdBy": "a530e865-23e8-4e2e-8020-e92c419a3867"
      }
    }
  ],
  "metadata": {}
}
```

为应用程序创建快照

您可以为特定应用程序创建快照。

开始之前

您必须具有要为其创建快照的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。

创建快照

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v1/apps/ {app_id} /appSaps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识要创建快照的受管应用程序。
JSON	body	是的。	提供快照的参数。请参见以下示例。

curl 示例：为应用程序创建快照

```
curl --request POST \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appSaps  
" \  
--include \  
--header "Content-Type: application/astra-appSnap+jso" \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN" \  
--data @JSONinput
```

JSON 输入示例

```
{  
  "type": "application/astra-appSnap",  
  "version": "1.1",  
  "name": "snapshot-david-1"  
}
```

为应用程序创建备份

您可以为特定应用程序创建备份、然后使用此备份还原或克隆此应用程序。

开始之前

您必须具有要备份的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。

创建备份

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v1/apps/ {app_id} /appBackups

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识要创建备份的应用程序。
JSON	body	是的。	提供备份的参数。请参见以下示例。

curl 示例：为应用程序创建备份

```
curl --request POST \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appBackups" \  
--include \  
--header "Content-Type: application/astra-appBackup+json" \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN" \  
--data @JSONinput
```

JSON 输入示例

```
{  
  "type": "application/astra-appBackup",  
  "version": "1.1",  
  "name": "backup-david-1"  
}
```

删除快照

您可以删除与应用程序关联的快照。

开始之前

您必须具备以下条件：

- 拥有快照的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。
- 要删除的快照的 ID 。如果需要，您可以使用此工作流 ["列出快照"](#) 以查找快照。

删除快照

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
删除	/accounts/ {account_id} /K8s/v1/apps/ {app_id} /appSnaps/ {appsnap_id}

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识拥有快照的受管应用程序。
Snapshot ID	路径	是的。	标识要删除的快照。

curl 示例：删除应用程序的单个快照

```
curl --request DELETE \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appSnaps  
/<SNAPSHOT_ID>" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

删除备份

您可以删除与应用程序关联的备份。


开始之前

您必须具备以下条件：

- 拥有备份的应用程序的ID。如果需要，您可以使用此工作流 ["列出应用程序"](#) 以查找应用程序。
- 要删除的备份的 ID 。如果需要，您可以使用此工作流 ["列出备份"](#) 以查找快照。

删除备份

执行以下 REST API 调用。



您可以使用可选的请求标头强制删除失败的备份，如下所述。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
删除	/accounts/ {account_id} /K8s/v1/apps/ {app_id} /appBackups/ {appBackup_id}

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
应用程序ID	路径	是的。	标识拥有备份的受管应用程序。
备份 ID	路径	是的。	标识要删除的备份。
强制删除	标题	否	用于强制删除失败的备份。

curl 示例：删除应用程序的单个备份

```
curl --request DELETE \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appBacku  
ps/<BACKUP_ID>" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

curl 示例：使用 **force** 选项删除应用程序的单个备份

```
curl --request DELETE \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/appBacku  
ps/<BACKUP_ID>" \  
--include \  
--header "Force-Delete: true" \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

创建数据保护策略

您可以根据一个或多个计划创建数据保护策略。

第1步：选择应用程序

执行工作流 ["列出应用程序"](#) 并选择所需的应用程序。

第2步：创建保护

执行以下REST API调用、为特定应用程序创建保护策略。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v1/apps/ {app_id} /计划

curl 示例

```
curl --request POST \  
--location  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v1/apps/$APP_ID/schedule  
s" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN" \  
--data @JSONinput
```

JSON 输入示例

```
{  
  "type": "application/astra-schedule",  
  "version": "1.3",  
  "name": "Backup Schedule",  
  "enabled": "true",  
  "granularity": "monthly",  
  "minute": "0",  
  "hour": "0",  
  "dayOfMonth": "1",  
  "snapshotRetention": "12",  
  "backupRetention": "12"  
}
```

克隆和还原应用程序

克隆应用程序

您可以通过克隆现有应用程序来创建新应用程序。

开始之前

请注意以下有关此工作流的信息：

- 未使用应用程序备份或快照
- 克隆操作在同一集群中执行
- 新应用程序将放置在其他命名空间中



要将应用程序克隆到其他集群，您需要根据环境需要更新 JSON 输入中的 `clusterid` 参数。

第1步：选择要克隆的应用程序

执行工作流 ["列出应用程序"](#) 并选择要克隆的应用程序。用于克隆应用程序的 REST 调用需要多个资源值。

第2步：克隆应用程序

执行以下 REST API 调用以克隆此应用程序。

HTTP 方法和端点

此 REST API 调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v2/apps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
JSON	body	是的。	提供克隆应用程序的参数。请参见以下示例。

curl 示例：克隆应用程序

```
curl --request POST \
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps" \
--include \
--header "Content-Type: application/astra-app+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data @JSONinput
```

JSON 输入示例

```
{
  "type": "application/astra-app",
  "version": "2.0",
  "name": "mysql-clone",
  "clusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "sourceClusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "namespace": "mysql-ns",
  "sourceAppID": "e591ee59-ea90-4a9f-8e6c-d2b6e8647096"
}
```


从快照克隆应用程序

您可以通过从快照克隆新应用程序来创建该应用程序。

开始之前

请注意以下有关此工作流的信息：

- 使用应用程序快照
- 克隆操作在同一集群中执行



要将应用程序克隆到其他集群，您需要根据环境需要更新 JSON 输入中的 clusterid 参数。

第1步：选择要克隆的应用程序

执行工作流 ["列出应用程序"](#) 并选择要克隆的应用程序。用于克隆应用程序的 REST 调用需要多个资源值。

第2步：选择要使用的快照

执行工作流 ["列出快照"](#) 并选择要使用的快照。

第3步：克隆应用程序

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v2/apps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
JSON	body	是的。	提供克隆应用程序的参数。请参见以下示例。

curl 示例：从快照克隆应用程序

```
curl --request POST \
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps" \
--include \
--header "Content-Type: application/astra-app+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data @JSONinput
```

JSON 输入示例

```
{
  "type": "application/astra-app",
  "version": "2.0",
  "name": "mysql-clone2",
  "clusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "sourceClusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "namespace": "mysql",
  "snapshotID": "e24515bd-a28e-4b28-b832-f3c74dbf32fb"
}
```

从备份克隆应用程序

您可以通过从备份克隆新应用程序来创建该应用程序。

开始之前

请注意以下有关此工作流的信息：

- 使用应用程序备份
- 克隆操作在同一集群中执行



要将应用程序克隆到其他集群，您需要根据环境需要更新 JSON 输入中的 clusterid 参数。

第1步：选择要克隆的应用程序

执行工作流 ["列出应用程序"](#) 并选择要克隆的应用程序。用于克隆应用程序的 REST 调用需要多个资源值。

第2步：选择要使用的备份

执行工作流 ["列出备份"](#) 并选择要使用的备份。

第3步：克隆应用程序

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
发布	/accounts/ {account_id} /k8s/v2/qpps

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
JSON	body	是的。	提供克隆应用程序的参数。请参见以下示例。

curl 示例：从备份克隆应用程序

```
curl --request POST \  
--location "https://astra.netapp.io/accounts/${ACCOUNT_ID}/k8s/v2/apps" \  
--include \  
--header "Content-Type: application/astra-app+json" \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN" \  
--data @JSONinput
```

JSON 输入示例

```
{  
  "type": "application/astra-app",  
  "version": "2.0",  
  "name": "mysql-clone3",  
  "clusterID": "30880586-d579-4d27-930f-a9633e59173b",  
  "sourceClusterID": "30880586-d579-4d27-930f-a9633e59173b",  
  "namespace": "mysql",  
  "backupID": "e24515bd-a28e-4b28-b832-f3c74dbf32fb"  
}
```

从备份还原应用程序

您可以通过从备份创建新应用程序来还原应用程序。

第1步：选择要还原的应用程序

执行工作流 ["列出应用程序"](#) 并选择要克隆的应用程序。用于还原应用程序的REST调用需要多个资源值。

第2步：选择要使用的备份

执行工作流 ["列出备份"](#) 并选择要使用的备份。

第3步：还原应用程序

执行以下 REST API 调用。您必须提供备份（如下所示）或快照的 ID 。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
PUT	/accounts/ {account_id} /k8s/v2/apps/ {app_id}

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
JSON	body	是的。	提供克隆应用程序的参数。请参见以下示例。

curl 示例：从备份原位还原应用程序

```
curl --request PUT \  
--location \  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/k8s/v2/apps/$APP_ID" \  
--include \  
--header "Content-Type: application/astra-app+json" \  
--header "ForceUpdate: true" \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN" \  
--data @JSONinput
```

JSON 输入示例

```
{  
  "type": "application/astra-app",  
  "version": "2.0",  
  "backupID": "e24515bd-a28e-4b28-b832-f3c74dbf32fb"  
}
```

命名空间

列出命名空间

您可以列出可用的命名空间。

执行以下REST API调用以列出命名空间。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
获取	/accounts/ {account_id} /topology/v1/namespaces

curl示例：返回所有命名空间的所有数据

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/namespaces" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

curl示例：返回所有命名空间的名称、状态和集群ID

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/namespaces?include=name,namespaceState,clusterID" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

JSON 输出示例

```
{  
  "items": [  
    [  
      "default",  
      "discovered",  
      "922f924a-a476-4a79-97f6-472571698154"  
    ],  
    [  
      "kube-node-lease",  
      "discovered",  
      "922f924a-a476-4a79-97f6-472571698154"  
    ],  
    [  
      "kube-public",  
      "discovered",  
      "922f924a-a476-4a79-97f6-472571698154"  
    ]  
  ]  
}
```

```

    ],
    [
        "kube-system",
        "discovered",
        "922f924a-a476-4a79-97f6-472571698154"
    ],
    [
        "mysql",
        "discovered",
        "922f924a-a476-4a79-97f6-472571698154"
    ],
    [
        "mysql-clone1",
        "discovered",
        "922f924a-a476-4a79-97f6-472571698154"
    ],
    [
        "netapp-acc-operator",
        "discovered",
        "922f924a-a476-4a79-97f6-472571698154"
    ],
    [
        "openshift",
        "discovered",
        "922f924a-a476-4a79-97f6-472571698154"
    ],
    [
        "trident",
        "discovered",
        "922f924a-a476-4a79-97f6-472571698154"
    ]
],
"metadata": {}
}

```

支持

列出通知

您可以列出特定 Astra 帐户的通知。您可以在监控系统活动或调试问题描述时执行此操作。

执行以下 REST API 调用。

HTTP方法和端点

此REST API调用使用以下方法和端点。

HTTP 方法	路径
获取	/accounts/ {account_id} /core/v1/notifications

其他输入参数

除了所有 REST API 调用通用的参数之外，此步骤的 curl 示例还使用以下参数。

参数	Type	Required	Description
筛选器	查询	否	也可以筛选要在响应中返回的通知。
包括	查询	否	也可以选择要在响应中返回的值。

curl 示例：返回所有通知

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/${ACCOUNT_ID}/core/v1/notifications" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

curl 示例：返回严重性为警告的通知的问题描述

```
curl --request GET \  
--location \  
"https://astra.netapp.io/accounts/${ACCOUNT_ID}/core/v1/notifications?filter=severity%20eq%20'warning'&include=description" \  
--include \  
--header "Accept: */*" \  
--header "Authorization: Bearer $API_TOKEN"
```

JSON 输出示例

```
{
  "items": [
    [
      "Trident on cluster david-ie-00 has failed or timed out;
installation of the Trident operator failed or is not yet complete;
operator failed to reach an installed state within 300.00 seconds;
container trident-operator not found in operator deployment"
    ],
    [
      "Trident on cluster david-ie-00 has failed or timed out;
installation of the Trident operator failed or is not yet complete;
operator failed to reach an installed state within 300.00 seconds;
container trident-operator not found in operator deployment"
    ]
  ],
  "metadata": {}
}
```

删除失败的应用程序

如果某个受管应用程序的备份或快照处于故障状态，您可能无法将其删除。在这种情况下，您可以使用下面所述的工作流手动删除此应用程序。

第1步：选择要删除的应用程序

执行工作流 ["列出应用程序"](#) 并选择要删除的应用程序。

第2步：列出应用程序的现有备份

执行工作流 ["列出备份"](#)。

第3步：删除所有备份

通过执行此工作流删除所有应用程序备份 ["删除备份"](#) 列表中的每个备份。

第4步：列出应用程序的现有快照

执行工作流 ["列出快照"](#)。

第5步：删除所有快照

执行工作流 ["删除快照"](#) 列表中的每个快照。

第6步：删除应用程序

执行工作流 ["取消管理应用程序"](#) 删除应用程序。

版权信息

版权所有 © 2025 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

本软件由 NetApp 按“原样”提供，不含任何明示或暗示担保，包括但不限于适销性以及针对特定用途的适用性的隐含担保，特此声明不承担任何责任。在任何情况下，对于因使用本软件而以任何方式造成的任何直接性、间接性、偶然性、特殊性、惩罚性或后果性损失（包括但不限于购买替代商品或服务；使用、数据或利润方面的损失；或者业务中断），无论原因如何以及基于何种责任理论，无论出于合同、严格责任或侵权行为（包括疏忽或其他行为），NetApp 均不承担责任，即使已被告知存在上述损失的可能性。

NetApp 保留在不另行通知的情况下随时对本文档所述的任何产品进行更改的权利。除非 NetApp 以书面形式明确同意，否则 NetApp 不承担因使用本文档所述产品而产生的任何责任或义务。使用或购买本产品不表示获得 NetApp 的任何专利权、商标权或任何其他知识产权许可。

本手册中描述的产品可能受一项或多项美国专利、外国专利或正在申请的专利的保护。

有限权利说明：政府使用、复制或公开本文档受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中“技术数据权利 — 非商用”条款第 (b)(3) 条规定的限制条件的约束。

本文档中所含数据与商业产品和/或商业服务（定义见 FAR 2.101）相关，属于 NetApp, Inc. 的专有信息。根据本协议提供的所有 NetApp 技术数据和计算机软件具有商业性质，并完全由私人出资开发。美国政府对这些数据的使用权具有非排他性、全球性、受限且不可撤销的许可，该许可既不可转让，也不可再许可，但仅限在与交付数据所依据的美国政府合同有关且受合同支持的情况下使用。除本文档规定的情形外，未经 NetApp, Inc. 事先书面批准，不得使用、披露、复制、修改、操作或显示这些数据。美国政府对国防部的授权仅限于 DFARS 的第 252.227-7015(b)（2014 年 2 月）条款中明确的权利。

商标信息

NetApp、NetApp 标识和 <http://www.netapp.com/TM> 上所列的商标是 NetApp, Inc. 的商标。其他公司和产品名称可能是其各自所有者的商标。