Astra REST实施

Astra Automation

NetApp
December 01, 2023

目录

Astra REST实施

核心设计

REST Web 服务

表述性状态传输（ Representational State Transfer ， REST ）是一种用于创建分布式 Web 应用程序的模式。在设计 Web 服务 API 时，它会建立一组用于公开基于服务器的资源并管理其状态的主流技术和最佳实践。虽然 REST 为应用程序开发提供了一致的基础，但每个 API 的详细信息可能因特定设计选项而异。在将 Astra Control REST API 用于实时部署之前，您应了解其特征。

资源和状态表示

资源是基于 Web 的系统的基本组件。创建 REST Web 服务应用程序时，早期设计任务包括：

- 识别系统或基于服务器的资源

  每个系统都使用和维护资源。资源可以是文件、业务事务、流程或管理实体。在设计基于 REST Web 服务的应用程序时，首先要完成的任务之一是识别资源。

- 资源状态和关联状态操作的定义

  资源始终处于数量有限的状态之一。必须明确定义状态以及用于影响状态更改的关联操作。

URI 端点

必须使用定义明确的寻址方案定义和提供每个 REST 资源。 资源所在的端点和标识的端点使用统一资源标识符（ Uniform Resource Identifier ， URI ）。URI 提供了一个通用框架，用于为网络中的每个资源创建唯一名称。统一资源定位器（ Uniform Resource Locator ， URL ）是一种用于 Web 服务的 URI 类型，用于标识和访问资源。资源通常以类似于文件目录的分层结构公开。

HTTP消息

超文本传输协议（ HTTP ）是 Web 服务客户端和服务器用来交换有关资源的请求和响应消息的协议。在设计 Web 服务应用程序时， HTTP 方法会映射到资源以及相应的状态管理操作。HTTP 为无状态。因此，要将一组相关请求和响应关联为一个事务的一部分，必须将追加信息包含在随请求和响应数据流一起提供的 HTTP 标头中。

JSON 格式化

虽然信息可以通过多种方式在 Web 服务客户端和服务器之间进行结构化和传输，但最受欢迎的选项是 JavaScript 对象表示法（ JSON ）。JSON 是一种行业标准，用于以纯文本形式表示简单数据结构，并用于传输描述资源的状态信息。Astra Control REST API 使用 JSON 格式化每个 HTTP 请求和响应正文中包含的数据。

资源和集合

通过 Astra Control REST API ，可以访问资源实例和资源实例集合。

> 从概念上讲， REST * 资源 * 类似于使用面向对象的编程（ OOP ）语言和系统定义的 * 对象 * 。有时，这些术语可以互换使用。但一般来说，在外部 REST API 环境中使用时，首选使用 "resource" ，而在服务器上存储的相应有状态实例数据中使用 "object" 。

Astra 资源的属性

Astra Control REST API 符合 RESTful 设计原则。每个 Astra 资源实例都是根据定义明确的资源类型创建的。一组相同类型的资源实例称为 * 集合 * 。API 调用会对单个资源或资源集合执行操作。

资源类型

Astra Control REST API 附带的资源类型具有以下特征：

- 每个资源类型均使用模式定义（通常在 JSON 中）
- 每个资源架构都包括资源类型和版本
- 资源类型在全局范围内是唯一的

资源实例

通过 Astra Control REST API 提供的资源实例具有以下特征：

- 资源实例是根据单个资源类型创建的
- 资源类型使用介质类型值来指示
- 实例由由 Astra 服务维护的有状态数据组成
- 每个实例均可通过一个唯一且长期存在的 URL 进行访问
- 如果某个资源实例可以具有多个表示形式，则可以使用不同的介质类型来请求所需的表示形式

资源收集

通过 Astra Control REST API 提供的资源收集具有以下特征：

- 一种资源类型的一组资源实例称为集合
- 资源集合具有一个唯一且长期存在的 URL

实例标识符

创建每个资源实例时，系统都会为其分配一个标识符。此标识符是一个 128 位 UUIDv4 值。分配的 UUIDv4 值是全局唯一且不可变的。发出创建新实例的API调用后、将向中的调用方返回具有关联ID的URL `Location` HTTP响应的标题。在引用资源实例时，您可以提取此标识符并在后续调用中使用它。

> 资源标识符是用于收集的主密钥。

Astra 资源的通用结构

每个 Astra Control 资源都使用一个通用结构进行定义。

通用数据

每个 Astra 资源都包含下表所示的键值。

密钥	Description
type	一种全局唯一资源类型，称为 * 资源类型 * 。
version	一种称为 * 资源版本 * 的版本标识符。
id	一种全局唯一标识符，称为 * 资源标识符 * 。
元数据	一个 JSON 对象，包含各种信息，包括用户和系统标签。

元数据对象

每个 Astra 资源附带的元数据 JSON 对象包含下表所示的键值。

密钥	Description
labels	与资源关联的客户端指定标签的 JSON 数组。
creationTimestamp	JSON 字符串，其中包含指示资源创建时间的时间戳。
modificationTimestamp	JSON 字符串，其中包含 ISO-8601 格式的时间戳，用于指示资源上次更改的时间。
已创建	JSON 字符串，其中包含创建资源的用户 ID 的 UIDv4 标识符。如果资源是由内部系统组件创建的，并且没有与创建实体关联的 UUID ，则使用 * 空 * UUID 。

资源状态

选定资源A `state` 用于编排生命周期过渡和控制访问的值。

HTTP 详细信息

Astra Control REST API使用HTTP以及相关参数对资源实例和集合执行操作。下面提供了HTTP 实施的详细信息。

API 事务和 CRUD 模型

Astra Control REST API 可实施一个事务模式，其中包含定义明确的操作和状态过渡。

请求和响应 API 事务

每次 REST API 调用都是作为对 Astra 服务的 HTTP 请求执行的。每个请求都会向客户端生成关联的响应。此请求响应对可视为 API 事务。

支持 CRUD 操作模式

通过 Astra Control REST API 提供的每个资源实例和集合均可根据 * CRU* 模型进行访问。有四个操作，每个操作都映射到一个 HTTP 方法。这些操作包括：

- 创建
- 读取
- 更新

- 删除

对于某些 Astra 资源，仅支持其中一部分操作。您应查看 "联机API参考" 有关特定 API 调用的详细信息。

HTTP 方法

下表显示了 API 支持的 HTTP 方法或动词。

方法	CRUD	Description
获取	读取	检索资源实例或集合的对象属性。在与集合结合使用时，此操作被视为 * 列表 * 操作。
发布	创建	根据输入参数创建新的资源实例。长期URL在中返回 `Location` 响应标头。
PUT	更新	使用提供的 JSON 请求正文更新整个资源实例。系统会保留用户不可修改的密钥值。
删除	删除	删除现有资源实例。

请求和响应标头

下表汇总了与 Astra Control REST API 一起使用的 HTTP 标头。

请参见 "RFC 7232" 和 "RFC 7233" 有关详细信息 …

标题	Type	使用说明
接受	请求	如果此值为"/"或未提供、 `application/json` 在Content-Type响应标头中返回。如果此值设置为 Astra 资源的介质类型，则内容类型标题中将返回相同的介质类型。
Authorization	请求	包含用户 API 密钥的承载令牌。
内容类型	响应	根据返回 `Accept` 请求标题。
ETAG	响应	随 RFC 7232 中定义的成功附带。该值是整个 JSON 资源的 MD5 值的十六进制表示形式。
如果匹配	请求	一个前提条件请求标头，如第 3.1 节 RFC 7232 中所述实施，并支持 * PUT * 请求。
if-Modified-since	请求	按照第3.4节RFC 7232中所述实施的前提条件请求标头，并支持*Put请求。
如果未修改，则从	请求	按照第3.4节RFC 7232中所述实施的前提条件请求标头，并支持*Put请求。
位置	响应	包含新创建资源的完整 URL 。

查询参数

以下查询参数可用于资源收集。请参见 "使用收集" 有关详细信息 …

查询参数	Description
包括	包含读取收集时应返回的字段。
筛选器	指示读取收集时要返回的资源必须匹配的字段。
订单	确定读取收集时返回的资源的排序顺序。
limit	限制读取集合时返回的最大资源数。
跳过	设置读取集合时要传递和跳过的资源数量。
count	指示是否应在元数据对象中返回资源总数。

HTTP status codes

下面介绍了 Astra Control REST API 使用的 HTTP 状态代码。

Astra Control REST API 还使用 * HTTP APIs* 标准的问题详细信息。请参见 "诊断和支持" 有关详细信息 …

代码	含义	Description
200	确定	表示未创建新资源实例的调用成功。
201	已创建	已成功创建对象，并且位置响应标头包含该对象的唯一标识符。
204.	No Content	尽管未返回任何内容，但请求成功。
400	请求错误	此请求输入无法识别或不适当。
401.	未授权	用户未获得授权，必须进行身份验证。
403.	已禁止	由于授权错误，访问被拒绝。
404	未找到	请求中引用的资源不存在。
409.	冲突	尝试创建对象失败，因为此对象已存在。
500	内部错误	服务器发生一般内部错误。
503.	服务不可用	由于某种原因，此服务尚未准备好处理此请求。

URL 格式

用于通过 REST API 访问资源实例或集合的 URL 的常规结构由多个值组成。此结构反映了底层对象模型和系统设计。

帐户作为 root

指向每个 REST 端点的资源路径的根是 Astra 帐户。因此、URL中的所有路径都以开头 `/account/{account_id}` 其中： `account_id` 是帐户的唯一UUIDv4值。内部结构这反映了一种设计，其中所有资源访问都基于特定帐户。

端点资源类别

Astra 资源端点分为三类：

- 核心 (`/core`)

- 受管应用程序 (`/k8s`)
- 拓扑 (`/topology`)

请参见 "Resources" 有关详细信息 …

类别版本

这三个资源类别中的每一个都有一个全局版本，用于控制所访问资源的版本。根据约定和定义、移动到资源类别的新主要版本(例如 `/v1` to `/v2`)将在API中引入中断更改。

资源实例或集合

根据是否访问资源实例或集合，可以在路径中使用资源类型和标识符的组合。

示例

- 资源路径

  根据上述结构、端点的典型路径为： `/accounts/{account_id}/core/v1/users`。

- 完整的 URL

  相应端点的完整URL为：
  `https://astra.netapp.io/accounts/{account_id}/core/v1/users`。

资源和端点

您可以访问通过Astra Control REST API提供的资源、以自动执行Astra部署。每个资源都可通过一个或多个端点来使用。下面介绍了可在自动化部署中使用的REST资源。

用于访问 Astra Control 资源的路径和完整 URL 的格式基于多个值。请参见 "URL 格式" 有关详细信息 …另请参见 "联机API参考" 有关使用 Astra 资源和端点的更多详细信息。

Astra Control REST 资源摘要

Astra Control REST API 中提供的主要资源端点分为三类。除了需要说明的情况外，可以使用一整套 CRUD 操作（创建，读取，更新，删除）来访问每个资源。

- 版本 * 列表示首次引入资源时的 Astra 版本。对于最近添加到REST API的资源、此字段为粗体。

核心资源

核心资源端点可提供建立和维护 Astra 运行时环境所需的基础服务。

资源	版本。	**Description**
Account	21.12.	通过帐户资源，您可以管理多租户 Astra Control 部署环境中的隔离租户。
ASUP	21.08.	ASUP 资源表示转发给 NetApp 支持部门的 AutoSupport 捆绑包。
证书	22.08.	证书资源表示已安装的用于对传出连接进行强身份验证的证书。

资源	版本。	Description
凭据	21.04.	凭据资源包含可用于 Astra 用户，集群，分段和存储后端的安全相关信息。
授权	21.08.	授权资源表示根据活动许可证和订阅可供帐户使用的功能和容量。
事件	21.04.	事件资源表示系统中发生的所有事件，包括归类为通知的子集。
执行钩	21.12.	执行钩资源表示自定义脚本，您可以在执行受管应用程序的快照之前或之后运行这些脚本。
功能	21.08.	这些功能资源表示选定的 Astra 功能，您可以查询这些功能来确定它们是否已在系统中启用。访问权限仅限于只读。
组	22.08.	组资源表示Astra组和关联资源。当前版本仅支持LDAP组。
挂钩源	21.12.	hook 源资源表示与执行 hook 一起使用的实际源代码。将源代码与执行控制分开具有多种优势，例如允许共享脚本。
LDAP组	22.11.	您可以列出已配置的LDAP服务器中的组。对LDAP组的访问是只读的。
LDAP用户	22.11.	您可以列出已配置的LDAP服务器中的用户。对LDAP用户的访问是只读的。
许可证	21.08.	许可证资源表示可用于 Astra 帐户的许可证。
通知	21.04.	通知资源表示具有通知目标的 Astra 事件。访问权限按用户提供。
软件包	22.04.	软件包资源用于注册和访问软件包定义。软件包由多个组件组成，包括文件，映像和其他项目。
权限	23.06.	权限资源表示与系统中的操作相关的权限。API提供对权限的只读访问权限。
Role	23.06.	角色资源表示系统中可用的角色。API提供对角色的只读访问权限。
角色绑定	21.04.	角色绑定资源表示特定用户对和帐户之间的关系。除了这两者之间的链接之外，还会通过特定角色为每个指定一组权限。
正在设置 …	21.08.	设置资源表示一组密钥值对，用于描述特定 Astra 帐户的功能。
订阅。	21.08.	订阅资源表示 Astra 帐户的活动订阅。
任务	22.11.	任务资源提供对受管任务的只读访问权限、并可用于显示内部长时间运行的任务的状态。
令牌	21.04.	令牌资源表示可通过编程方式访问 Astra Control REST API 的令牌。
未读通知	21.04.	未读通知资源表示已分配给特定用户但尚未读取的通知。
升级	22.04.	通过升级资源，您可以访问软件组件并启动升级。
用户	21.04.	用户资源表示 Astra 用户能够根据其定义的角色访问系统。

受管应用程序资源

通过受管应用程序资源端点，可以访问受管 Kubernetes 应用程序。

资源	版本。	Description
应用程序资产	21.04.	应用程序资产资源表示管理 Astra 应用程序所需的内部状态信息集合。
应用程序备份	21.04.	应用程序备份资源表示受管应用程序的备份。
应用程序快照	21.04.	应用程序快照资源表示受管应用程序的快照。

资源	版本。	Description
执行挂机覆盖	21.12.	使用执行挂钩覆盖资源，您可以根据需要为特定应用程序禁用预加载的 NetApp 默认执行挂钩。
计划	21.04.	计划资源表示在数据保护策略中为受管应用程序计划的数据保护操作。

拓扑资源

通过拓扑资源端点可以访问非受管应用程序和存储资源。

资源	版本。	Description
API资源	22.11.	API资源端点提供对特定受管集群中Kubernetes资源的只读访问权限。
应用程序	21.04.	应用程序资源表示所有 Kubernetes 应用程序，包括那些不受 Astra 管理的应用程序。
AppMirror	22.08.	AppMirror资源表示用于管理应用程序镜像关系的AppMirror资源。
存储分段	21.08.	存储分段资源表示用于存储由 Astra 管理的应用程序备份的 S3 云分段。
云	21.08.	云资源表示 Astra 客户端为管理集群和应用程序而可以连接到的云。
集群	21.08.	集群资源表示不受 Kubernetes 管理的 Kubernetes 集群。
集群节点	21.12.	集群节点资源允许您访问 Kubernetes 集群中的各个节点，从而提供额外的解析。
受管集群	21.08.	受管集群资源表示当前由 Kubernetes 管理的 Kubernetes 集群。
命名空间	21.12.	命名空间资源用于访问 Kubernetes 集群中使用的命名空间。
存储后端	21.08.	存储后端资源表示可由 Astra 管理的集群和应用程序使用的存储服务提供商。
存储类	21.08.	存储类资源表示已发现并可供特定受管集群使用的不同存储类或类型。
Volume	21.04.	卷资源表示与受管应用程序关联的 Kubernetes 存储卷。

其他资源和端点

您可以使用多种其他资源和端点来支持 Astra 部署。

这些资源和端点当前未包含在 Astra Control REST API 参考文档中。

OpenAPI

通过 OpenAPI 端点可以访问当前的 OpenAPI JSON 文档和其他相关资源。

OpenMetrics

通过 OpenMetrics 端点，您可以通过 OpenMetrics 资源访问帐户指标。Astra 控制中心部署模式支持此功能。

其他注意事项

RBAC安全性

Astra REST API支持基于角色的访问控制(Role-Based Access Control、RBAC)来授予和限制对系统功能的访问权限。

Astra 角色

每个 Astra 用户都分配有一个角色，用于确定可执行的操作。这些角色按层次结构进行排列，如下表所述。

Role	Description
所有者	具有管理员角色的所有权限，并且还可以删除 Astra 帐户。
管理员	具有成员角色的所有权限，并且还可以邀请用户加入帐户。
成员	可以全面管理 Astra 应用程序和计算资源。
查看器	仅限查看资源。

具有命名空间粒度的增强型 RBAC

此功能是在 Astra REST API 22.04 版中引入的。

为特定用户建立角色绑定后，可以应用一个限制来限制用户有权访问的命名空间。可通过多种方法定义此限制，如下表所述。请参见参数 `roleContraints` 有关详细信息、请参见角色绑定API。

命名空间	Description
全部	用户可以通过通配符参数 "*" 访问所有命名空间。这是保持向后兼容性的默认值。
无	尽管约束列表为空，但仍会指定此限制列表。这表示用户无法访问任何命名空间。
命名空间列表	包含命名空间的 UUID ，这会将用户限制为单个命名空间。此外，还可以使用逗号分隔列表来访问多个命名空间。
Label	指定了一个标签，并允许访问所有匹配的命名空间。

使用收集

Astra Control REST API 提供了多种不同的方法来通过定义的查询参数访问资源收集。

选择值

您可以使用为每个资源实例指定应返回的键值对 `include` 参数。所有实例都会在响应正文中返回。

筛选

通过收集资源筛选， API 用户可以指定条件，以确定是否在响应正文中返回资源。。 `filter` 参数用于指示筛选条件。

排序

通过收集资源排序， API 用户可以在响应正文中指定资源的返回顺序。。 `orderBy` 参数用于指示筛选条件。

分页

您可以通过使用限制为请求返回的资源实例数来强制执行分页 `limit` 参数。

计数

如果包括布尔参数 `count` 设置为 `true`、给定响应的返回数组中的资源数量在元数据部分提供。

诊断和支持

Astra Control REST API 提供了多种支持功能，可用于诊断和调试。

API resources

API 资源提供了多种可提供诊断信息和支持的 Astra 功能。

Type	Description
事件	在 Astra 处理过程中记录的系统活动。
通知	被视为足够重要的一小部分事件，可提供给用户。
未读通知	用户尚未读取或检索的通知。

撤消 API 令牌

您可以在 Astra Web 界面上撤消不再需要的 API 令牌。

开始之前

您需要凭据才能登录到适用于您的部署的ASRA Web用户界面。 您还应确定要撤消的令牌。

关于此任务

令牌撤消后，它将立即永久不可用。

步骤

1. 使用您的帐户凭据登录ASRA、如下所示：
   a. Asta控制服务： "https://astra.netapp.io"
   b. Astra Control Center：使用安装期间为您的本地环境建立的URL
2. 单击页面右上角的图图标并选择 * API access* 。
3. 选择要撤消的一个或多个令牌。
4. 在 * 操作 * 下拉框下，单击 * 撤消令牌 * 。