



À l'aide de Python

Astra Automation 22.04

NetApp
December 04, 2023

Sommaire

- À l'aide de Python 1
 - Kit de développement logiciel NetApp Astra Control Python 1
 - Python natif 2

À l'aide de Python

Kit de développement logiciel NetApp Astra Control Python

Le kit de développement logiciel NetApp Astra Control Python est un logiciel open source qui permet d'automatiser le déploiement d'Astra Control. Il constitue également une ressource précieuse pour en savoir plus sur l'API REST Astra Control, peut-être dans le cadre de la création de votre propre plateforme d'automatisation.



Dans un souci de simplicité, le kit de développement NetApp Astra Control Python sera appelé **SDK** au cours du reste de cette page.

Deux outils logiciels connexes

Le SDK inclut deux outils différents bien que associés qui fonctionnent à différents niveaux d'abstraction lors de l'accès à l'API REST Astra Control.

Kit de développement Astra

Le kit de développement logiciel Astra fournit les principales fonctionnalités de la plateforme. Il inclut un ensemble de classes Python qui abstrait les appels de l'API REST sous-jacente. Ces classes prennent en charge les actions administratives sur diverses ressources Astra Control, notamment les applications, les sauvegardes, les instantanés et les clusters.

Le kit de développement Astra fait partie de l'offre et est fourni en une seule pièce `astraSDK.py` fichier. Vous pouvez importer ce fichier dans votre environnement et utiliser les classes directement.



Le **SDK Python de contrôle Astra** (ou uniquement SDK) de NetApp est le nom de l'ensemble du package. Le **Astra SDK** fait référence aux classes Python de base dans le fichier unique `astraSDK.py`.

Script Toolkit

En plus du fichier Astra SDK, le `toolkit.py` script également disponible. Ce script fonctionne à un niveau d'abstraction plus élevé en donnant accès à des actions administratives discrètes définies en interne comme des fonctions Python. Le script importe le SDK Astra et fait des appels aux classes selon les besoins.

Comment y accéder

Vous pouvez accéder au SDK de la manière suivante.

Pack Python

Le SDK est disponible à l'adresse "[Index des paquets Python](#)" sous le nom **netapp-astra-toolkits**. Un numéro de version est attribué au package et continuera d'être mis à jour au besoin. Vous devez utiliser l'utilitaire de gestion de paquets **PIP** pour installer le package dans votre environnement.

Voir "[PyPI : kit de développement Python de contrôle NetApp Astra](#)" pour en savoir plus.

Code source GitHub

Le code source du SDK est également disponible sur GitHub. Le référentiel inclut les éléments suivants :

- `astraSDK.py` (SDK Astra avec classes Python)
- `toolkit.py` (script basé sur les fonctions de niveau supérieur)
- Conditions requises et instructions détaillées pour l'installation
- Scripts d'installation
- Documentation complémentaire

Vous pouvez cloner le "[GitHub : kits NetApp/netapp-astra-toolkits](#)" référentiel pour votre environnement local.

Conditions requises pour l'installation et de base

Il y a plusieurs options et conditions requises à considérer dans le cadre de l'installation de l'emballage et de la préparation à l'utilisation.

Résumé des options d'installation

Vous pouvez installer le SDK de l'une des manières suivantes :

- Utilisez PIP pour installer le package de PyPI dans votre environnement Python
- Clonez le référentiel Git Hub et :
 - Déploiement du package en tant que conteneur Docker (qui inclut tout ce dont vous avez besoin)
 - Copiez les deux fichiers Python de base afin qu'ils soient accessibles à votre code client Python

Reportez-vous aux pages PyPI et GitHub pour plus d'informations.

Exigences relatives à l'environnement Astra Control

Qu'il s'agisse d'utiliser directement les classes Python du SDK Astra ou des fonctions du `toolkit.py` À terme, vous accéderez à l'API REST lors d'un déploiement d'Astra Control. Vous aurez donc besoin d'un compte Astra et d'un jeton API. Voir "[Avant de commencer](#)" Et les autres pages de la section **Get Started** de cette documentation pour plus d'informations.

Exigences relatives au kit de développement logiciel NetApp Astra Control Python

Le SDK a plusieurs conditions préalables liées à l'environnement Python local. Par exemple, vous devez utiliser Python 3.5 ou ultérieur. En outre, plusieurs paquets Python sont requis. Pour plus d'informations, consultez la page de référentiel GitHub ou la page de package PyPI.

Résumé des ressources utiles

Voici quelques ressources utiles pour commencer.

- "[PyPI : kit de développement Python de contrôle NetApp Astra](#)"
- "[GitHub : kits NetApp/netapp-astra-toolkits](#)"

Python natif

Avant de commencer

Python est un langage de développement populaire en particulier pour l'automatisation des data centers. Avant d'utiliser les fonctions natives de Python avec plusieurs packages courants, vous devez préparer l'environnement et les fichiers d'entrée requis.



En plus d'accéder directement à l'API REST Astra Control avec Python, NetApp propose un kit d'outils qui résume l'API et élimine une partie de la complexité. Voir "[Kit de développement logiciel NetApp Astra Control Python](#)" pour en savoir plus.

Préparation de l'environnement

Les exigences de configuration de base pour exécuter les scripts Python sont décrites ci-dessous.

Python 3

La dernière version de Python 3 doit être installée.

Bibliothèques supplémentaires

Les bibliothèques **requêtes** et **urllib3** doivent être installées. Vous pouvez utiliser pip ou un autre outil de gestion Python, selon les besoins de votre environnement.

Accès réseau

Le poste de travail sur lequel les scripts s'exécutent doit disposer d'un accès réseau et pouvoir accéder à Astra Control. Lorsque vous utilisez le service Astra Control, vous devez être connecté à Internet et être en mesure de vous connecter au service à <https://astra.netapp.io>.

Informations d'identité

Vous avez besoin d'un compte Astra valide avec l'identifiant de compte et le jeton API. Voir "[Obtenir un jeton API](#)" pour en savoir plus.

Créez les fichiers d'entrée JSON

Les scripts Python s'appuient sur les informations de configuration contenues dans les fichiers d'entrée JSON. Des exemples de fichiers sont fournis ci-dessous.



Vous devez mettre à jour les échantillons en fonction de votre environnement.

Informations d'identité

Le fichier suivant contient le jeton API et le compte Astra. Vous devez transmettre ce fichier aux scripts Python à l'aide de `-i` (ou `--identity`) Paramètre CLI.

```
{
  "api_token": "kH4CA_uVIa8q9UuPzhJaAHaGlaR7-no901DkkrVjIXk=",
  "account_id": "5131dfdf-03a4-5218-ad4b-fe84442b9786"
}
```

Répertorier les applications gérées

Vous pouvez utiliser le script suivant pour répertorier les applications gérées pour votre compte Astra.



Voir "[Avant de commencer](#)" Par exemple le fichier d'entrée JSON requis.

```
#!/usr/bin/env python3
##-----
-----
#
# Usage: python3 list_man_apps.py -i identity_file.json
#
# (C) Copyright 2021 NetApp, Inc.
#
# This sample code is provided AS IS, with no support or warranties of
# any kind, including but not limited for warranties of merchantability
# or fitness of any kind, expressed or implied. Permission to use,
# reproduce, modify and create derivatives of the sample code is granted
# solely for the purpose of researching, designing, developing and
# testing a software application product for use with NetApp products,
# provided that the above copyright notice appears in all copies and
# that the software application product is distributed pursuant to terms
# no less restrictive than those set forth herein.
#
##-----
-----

import argparse
import json
import requests
import urllib3
import sys

# Global variables
api_token = ""
account_id = ""

def get_managed_apps():
    ''' Get and print the list of managed apps '''

    # Global variables
    global api_token
    global account_id

    # Create an HTTP session
    sess1 = requests.Session()

    # Suppress SSL unsigned certificate warning
    urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

    # Create URL
    url1 = "https://astra.netapp.io/accounts/" + account_id +
```

```

"/k8s/v1/managedApps"

# Headers and response output
req_headers = {}
resp_headers = {}
resp_data = {}

# Prepare the request headers
req_headers.clear
req_headers['Authorization'] = "Bearer " + api_token
req_headers['Content-Type'] = "application/astra-managedApp+json"
req_headers['Accept'] = "application/astra-managedApp+json"

# Make the REST call
try:
    resp1 = sess1.request('get', url1, headers=req_headers,
allow_redirects=True, verify=False)

except requests.exceptions.ConnectionError:
    print("Connection failed")
    sys.exit(1)

# Retrieve the output
http_code = resp1.status_code
resp_headers = resp1.headers

# Print the list of managed apps
if resp1.ok:
    resp_data = json.loads(resp1.text)
    items = resp_data['items']
    for i in items:
        print(" ")
        print("Name: " + i['name'])
        print("ID: " + i['id'])
        print("State: " + i['state'])
    else:
        print("Failed with HTTP status code: " + str(http_code))

print(" ")

# Close the session
sess1.close()

return

def read_id_file(idf):
    ''' Read the identity file and save values '''

```

```

# Global variables
global api_token
global account_id

with open(idf) as f:
    data = json.load(f)

api_token = data['api_token']
account_id = data['account_id']

return

def main(args):
    ''' Main top level function '''

    # Global variables
    global api_token
    global account_id

    # Retrieve name of JSON input file
    identity_file = args.id_file

    # Get token and account
    read_id_file(identity_file)

    # Issue REST call
    get_managed_apps()

    return

def parseArgs():
    ''' Parse the CLI input parameters '''

    parser = argparse.ArgumentParser(description='Astra REST API -
List the managed apps',
                                    add_help = True)
    parser.add_argument("-i", "--identity", action="store", dest
="id_file", default=None,
                        help='(Req) Name of the identity input file',
                        required=True)

    return parser.parse_args()

if __name__ == '__main__':
    ''' Begin here '''

```



```
# Parse input parameters
args = parseArgs()

# Call main function
main(args)
```

Informations sur le copyright

Copyright © 2023 NetApp, Inc. Tous droits réservés. Imprimé aux États-Unis. Aucune partie de ce document protégé par copyright ne peut être reproduite sous quelque forme que ce soit ou selon quelque méthode que ce soit (graphique, électronique ou mécanique, notamment par photocopie, enregistrement ou stockage dans un système de récupération électronique) sans l'autorisation écrite préalable du détenteur du droit de copyright.

Les logiciels dérivés des éléments NetApp protégés par copyright sont soumis à la licence et à l'avis de non-responsabilité suivants :

CE LOGICIEL EST FOURNI PAR NETAPP « EN L'ÉTAT » ET SANS GARANTIES EXPRESSES OU TACITES, Y COMPRIS LES GARANTIES TACITES DE QUALITÉ MARCHANDE ET D'ADÉQUATION À UN USAGE PARTICULIER, QUI SONT EXCLUES PAR LES PRÉSENTES. EN AUCUN CAS NETAPP NE SERA TENU POUR RESPONSABLE DE DOMMAGES DIRECTS, INDIRECTS, ACCESSOIRES, PARTICULIERS OU EXEMPLAIRES (Y COMPRIS L'ACHAT DE BIENS ET DE SERVICES DE SUBSTITUTION, LA PERTE DE JOUISSANCE, DE DONNÉES OU DE PROFITS, OU L'INTERRUPTION D'ACTIVITÉ), QUELLES QU'EN SOIENT LA CAUSE ET LA DOCTRINE DE RESPONSABILITÉ, QU'IL S'AGISSE DE RESPONSABILITÉ CONTRACTUELLE, STRICTE OU DÉLICTELLE (Y COMPRIS LA NÉGLIGENCE OU AUTRE) DÉCOULANT DE L'UTILISATION DE CE LOGICIEL, MÊME SI LA SOCIÉTÉ A ÉTÉ INFORMÉE DE LA POSSIBILITÉ DE TELS DOMMAGES.

NetApp se réserve le droit de modifier les produits décrits dans le présent document à tout moment et sans préavis. NetApp décline toute responsabilité découlant de l'utilisation des produits décrits dans le présent document, sauf accord explicite écrit de NetApp. L'utilisation ou l'achat de ce produit ne concède pas de licence dans le cadre de droits de brevet, de droits de marque commerciale ou de tout autre droit de propriété intellectuelle de NetApp.

Le produit décrit dans ce manuel peut être protégé par un ou plusieurs brevets américains, étrangers ou par une demande en attente.

LÉGENDE DE RESTRICTION DES DROITS : L'utilisation, la duplication ou la divulgation par le gouvernement sont sujettes aux restrictions énoncées dans le sous-paragraphe (b)(3) de la clause Rights in Technical Data-Noncommercial Items du DFARS 252.227-7013 (février 2014) et du FAR 52.227-19 (décembre 2007).

Les données contenues dans les présentes se rapportent à un produit et/ou service commercial (tel que défini par la clause FAR 2.101). Il s'agit de données propriétaires de NetApp, Inc. Toutes les données techniques et tous les logiciels fournis par NetApp en vertu du présent Accord sont à caractère commercial et ont été exclusivement développés à l'aide de fonds privés. Le gouvernement des États-Unis dispose d'une licence limitée irrévocable, non exclusive, non cessible, non transférable et mondiale. Cette licence lui permet d'utiliser uniquement les données relatives au contrat du gouvernement des États-Unis d'après lequel les données lui ont été fournies ou celles qui sont nécessaires à son exécution. Sauf dispositions contraires énoncées dans les présentes, l'utilisation, la divulgation, la reproduction, la modification, l'exécution, l'affichage des données sont interdits sans avoir obtenu le consentement écrit préalable de NetApp, Inc. Les droits de licences du Département de la Défense du gouvernement des États-Unis se limitent aux droits identifiés par la clause 252.227-7015(b) du DFARS (février 2014).

Informations sur les marques commerciales

NETAPP, le logo NETAPP et les marques citées sur le site <http://www.netapp.com/TM> sont des marques déposées ou des marques commerciales de NetApp, Inc. Les autres noms de marques et de produits sont des marques commerciales de leurs propriétaires respectifs.