iam

index
/python_sdk/iam.py

Modules

base64
exceptions
hashlib
hmac
json
os
pprint
requests
six
sys
time
urllib
six.moves.urllib
xmltodict
yaml

Functions

assign_user_to_group(user_name, group_name)

```
Assign a user to a group in your JCS account.

Parameters:

    user_name (string)
    The name of user to be assigned to the group.

    group_name (string)
    The name of the group, the user should be assigned to.
```

attach_policy_to_group(policy_name, group_name)

```
Attaches a specified user based policy to a specified JCS IAM group. You must have the AttachPolicyToGroup permissions to run thi

Parameters:

    policy_name (string)
    The policy name of the user based policy to be attached.

    group_name (string)
    The Group Name of the group to which policy needs to be attached.
```

attach_policy_to_resource(resource, policy_name)

```
Attaches a resource based policy to a list of resources.

Parameters:

    resource(resource json)
    The resource jrns to be attached to the policy.

    policy_name (string)
    The policy name of the resource based policy to be attached.
```

attach_policy_to_user(policy_name, user_name)

```
Attaches a specified user based policy to a specified user. You must have the AttachPolicyToUser permissions to run this request.

Parameters:

    policy_name (string)
    The policy name of the user based policy to be attached.

    user_name (string)
    The User Name of the user from whom policy needs to be attached.
```

check_user_in_group(user_name, group_name)

```
Check if the user belongs to a group in your JCS account.

Parameters:

    user_name (string)
    The name of user to be checked in the group.

    group_name (string)
    The name of the group, the user should be checked to.
```

create_credential(user_name)

```
Creates Access Key/Secret Key pair for a new user.

Parameters:

    user_name (string)
    The name of the user to create credential for.
```

create_group(name, description=None)

```
Creates a new group for your JCS account.

Parameters:

    name (string)
    The name of the group to create.

    description (string)
    description for the group.
    This parameter is optional.
```

create_policy(policy_document)

```
Creates a new policy for your JCS account.

Parameters:

    policy_document (string)
    The policy document for the policy
```

create_resource_based_policy(policy_document)

```
Creates a resource based policy in your JCS account.

Parameters:

    policy-document (Policy Json)
    The Url Encoded Policy Document  for a resource based policy.
```

create_user(name, password=None, email=None)

```
Creates a new user for your JCS account.

Parameters:

    name (string)
    The name of the user to create.

    password (string)
    Assign a password to the user.
    This parameter is optional.

    email (string)
    The email id of the user.
    This paramater is optional.
```

create_virtual_mfa_device(name)

```
Creates a virtual mfa device in the caller's account.

Parameters:

    name (string)
    The name of the virtual mfa device.
```

deactivate_mfa_device(device_name, user_name=None)

```
Deactivates an mfa device attached to a user.

Parameters:

    device_name (string)
    Name of the device to be deactivated.

    user_name (string)
    Name of the user whose device is to be deactivated.
```

delete_credential(access_key=None, id=None)

```
Deletes Access Key/Secret Key pair for a user.

Parameters:

    access-key (string)
    The access-key corresponding to the crednetial to be deleted.

    id (string)
    The id of the credential to be deleted.
    This parameter is optional.

    Either of access-key or id is required.
```

delete_group(name)

```
Deletes the group in your JCS account.

Parameters:

    name (string)
    The name of the group to delete.
```

delete_policy(name)

```
Deletes a policy for your JCS account.

Parameters:

    name (string)
    The name of the policy.
```

delete_resource_based_policy(name)

```
Deletes the resource based policy in your JCS account.

Parameters:

    name (string)
    The name of the policy which is to be deleted.
```

delete_user(name)

```
Deletes a specified user.

Parameters:

    name (string)
    The name of the user to delete.
```

delete_virtual_mfa_device(name)

```
Delete a virtual mfa device.

Parameters:

    name (string)
    Name of the mfa device to be deleted.
```

detach_policy_from_group(policy_name, group_name)

```
Detaches a specified user based policy from a specified group. You must have the DetachPolicyFromGroup permissions to run this re

Parameters:

    policy_name (string)
    The policy name of the user based policy to be detached.

    group_name (string)
    The group Name of the group from which policy needs to be detached.
```

detach_policy_from_resource(resource, policy_name)

```
Detaches a resource based policy from a list of resources.

Parameters:

    resource(resource json)
    The resource jrns to be detached from the policy.

    policy_name (string)
    The policy name of the resource based policy to be detached.
```

detach_policy_from_user(policy_name, user_name)

```
Detaches a specified user based policy from a specified user. You must have the DetachPolicyFromUser permissions to run this requ

Parameters:

    policy_name (string)
    The policy name of the user based policy to be detached.

    user_name (string)
    The User Name of the user from whom policy needs to be detached.
```

enable_mfa_device(user_name, device_name, code1, code2)

```
Attaches an mfa device to a user. There can be at most one mfa device attached to a user.

Parameters:

    user_name (string)
    User which needs to be mfa enabled.

    device_name (string)
    Name of the device to be attached to the user.

    code1
    OTP code, last but one.

    code2
    OTP code now.
```

get_group(name)

```
Retrieves information for a specified group for your JCS account.

Parameters:

    name (string)
    The name of the group to retrieve information for.
```

get_group_summary(name)

```
Returns the summary of the specified group.

Parameters:

    name (string)
    Name of the group for which summary needs to be returned.
```

get_policy(name)

```
Get a policy for your JCS account.

Parameters:

    name (string)
    The name of the policy.
```

get_policy_summary(name)

```
Retrieves summary information about a specified policy including the attached _entities, about a specified JCS IAM user based pol

Parameters:

    name (string)
    The name of the policy whose summary to be fetched.
```

get_resource_based_policy(name)

```
Retreives details about a specific resource based policy in your JCS account.

Parameters:
    name (string)
    The name of the policy whose information is to be fetched.
```

get_resource_based_policy_summary(name)

```
Retrieves summary information about a specified resource based policy including the attached resources.

Parameters:

    name (string)
    The name of the resource based policy whose summary is to be fetched.
```

get_user(name)

```
Retrieves information for a specified user in your JCS account.

Parameters:

    name (string)
    The name of the user to retrieve information for.
```

get_user_credential(user_name)

```
Retrieves access keys for a user.

Parameters:

    user_name (string)
    The name of the user to retrieve credentials for.
```

get_user_summary(name)

```
Returns the summary of the specified user.

Parameters:

    name (string)
    Name of the user for which summary needs to be returned.
```

list_groups()

```
List the details of all the group in your JCS account.

Parameters:

    No parameters.
```

list_groups_for_user(name)

```
List groups for the user. You must have the permission ListGroupsForUser permssion to run this request.

Parameters:

    name (string)
    The name of the user for which groups needs to be listed
```

list_policies()

```
Lists all policies for your JCS account.

Parameters:

    No parameters.
```

list_resource_based_policies()

```
List the details of all the resource based policies in your JCS account.

Parameters:

    No parameters
```

list_user_in_group(name)

```
Lists users attached to a group

Parameters:

    name (string)
    The name of the group for which users needs to be listed
```

list_users()

```
List the details of all the users in your JCS account.

Parameters:

    No parameters.
```

list_virtual_mfa_devices(assign_status)

```
Lists all virtual mfa devices in an account based on assign_status

Parameters:

    assign_status
```

remove_user_from_group(user_name, group_name)

```
Remove the user from a group in your JCS account.

Parameters:

    user-name (string)
    The name of user to be removed from the group.
    group-name (string)
    The name of the group, the user should be removed from.
```

resync_mfa_device(device_name, code1, code2, user_name=None)

```
Resyncs an MFA device which got disabled due to three wrong otp attempts.

Parameters:

    device_name (string)
    Name of the device that requires a resync.

    code1
    OTP code, last but one.

    code2
    OTP code now.

    user_name (string)
    Name of the user whose device needs resync.
```

update_group(name, new_description)

```
Updates the name and description of a group.

Parameters:

    name (string)
    The name of the group which needs to be update. Atleast one of --id or --name should be specified.

    new_description(string)
    New description of the group.
```

update_policy(name, policy_document)

```
Updates an existing policy for your JCS account.

Parameters:

    name (string)
    The name of the policy.
```

update_user(name, new_password=None, new_email=None)

```
Updates the email_id or password of a user.

Parameters:

    name (string)
    The name of the user which needs to be updated.

    new_email (string)
    New email of the user

    new_password (string)
    New password of the user.

    Alreast one of new_email or new_password needs to be specified.
```

Data