



# GetUserLinkedAccount

## SDK Type: Javascript/Node.js

**Description** Gets an Asset Issuer account for a User.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.getUserLinkedAccount(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetUserInventoryWithPayload

## SDK Type: Javascript/Node.js

**Description** Get's a users inventory with payload data.

### Required Inputs

- **asset\_issuer\_uid** ~ \_
- **user\_uid** ~ \_

### Notes

*Prerequisites are Listed below, if any are required.*

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.getUserInventoryWithPayload(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateMarketplaceAssetUnlimited

## SDK Type: Javascript/Node.js

**Description** Create's a Marketplace Asset from a Asset Schema.

### Required Inputs

- **asset\_schema\_uid** ~ The UID of an Asset Schema, which is defined by the unique properties that make up a digital asset.
- **json\_payload** ~ The JSON Schema Payload, which is required to mint an Asset.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var json_payload = "";
sdk.createMarketplaceAssetUnlimited(asset_schema_uid, json_payload).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateMarketplaceAssetUnlimitedWithImage

## SDK Type: Javascript/Node.js

**Description** Create's a Marketplace Asset from a Asset Schema.

### Required Inputs

- **asset\_schema\_uid** ~ The UID of an Asset Schema, which is defined by the unique properties that make up a digital asset.
- **json\_payload** ~ The JSON Schema Payload, which is required to mint an Asset.
- **image\_file** ~ Directory location of an image file.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var json_payload = "";
var image_file = "";
sdk.createMarketplaceAssetUnlimitedWithImage(asset_schema_uid, json_payload, image_file).then((data)
=> {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateMarketplaceAssetQuantity

## SDK Type: Javascript/Node.js

**Description** Create's a Marketplace Asset from a Asset Schema.

### Required Inputs

- **asset\_schema\_uid** ~ *The UID of an Asset Schema, which is defined by the unique properties that make up a digital asset.*
- **json\_payload** ~ *The JSON Schema Payload, which is required to mint an Asset.*
- **quantity** ~ *This is the Quantity of the sellable amount.*

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var json_payload = "";
var quantity = "";
sdk.createMarketplaceAssetQuantity(asset_schema_uid, json_payload, quantity).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateMarketplaceAssetQuantitydWithImage

## SDK Type: Javascript/Node.js

**Description** Create's a Marketplace Asset from a Asset Schema.

### Required Inputs

- **asset\_schema\_uid** ~ The UID of an Asset Schema, which is defined by the unique properties that make up a digital asset.
- **json\_payload** ~ The JSON Schema Payload, which is required to mint an Asset.
- **image\_file** ~ Directory location of an image file.
- **quantity** ~ This is the Quantity of the sellable amount.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **CreateAssetSchema**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var json_payload = "";
var image_file = "";
var quantity = "";
sdk.createMarketplaceAssetQuantitydWithImage(asset_schema_uid, json_payload, image_file,
quantity).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateMarketplaceListing

## SDK Type: Javascript/Node.js

**Description** Assign's a Marketplace Asset to be Listed on the Marketplace.

### Required Inputs

- **user\_inventory\_uid** ~ \_\_
- **sale\_price** ~ \_\_
- **listing\_data** ~ \_\_

### Notes

*Prerequisites are Listed below, if any are required.*

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)
4. [CreateMarketplaceAssetQuantity](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var user_inventory_uid = "";
var sale_price = "";
var listing_data = "";
sdk.createMarketplaceListing(user_inventory_uid, sale_price, listing_data).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateUserAssetIssuerOwnership

## SDK Type: Javascript/Node.js

**Description** Links ownership of an existing User to an Asset Issuer.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.
- **game\_username** ~ The username from an Asset Issuer's database to link to a User.
- **game\_uid** ~ The Unique Identifier from an Asset Issuer's database to link a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
var game_username = "";
var game_uid = "";
sdk.createUserAssetIssuerOwnership(asset_issuer_uid, user_uid, game_username, game_uid).then((data)
=> {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UpdateUserAssetIssuerOwnership

## SDK Type: Javascript/Node.js

**Description** Updates ownership of an existing User to an Asset Issuer.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.
- **game\_username** ~ The username from an Asset Issuer's database to link to a User.
- **game\_uid** ~ The Unique Identifier from an Asset Issuer's database to link a User.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
var game_username = "";
var game_uid = "";
sdk.updateUserAssetIssuerOwnership(asset_issuer_uid, user_uid, game_username, game_uid).then((data)
=> {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# DeleteUserAssetIssuerOwnership

## SDK Type: Javascript/Node.js

**Description** Removes the ownership link between an Asset Issuer & a User.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.deleteUserAssetIssuerOwnership(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# DispatchExternalInvite

## SDK Type: Javascript/Node.js

**Description** Invites Users to join CONTACT Systems.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **game\_email** ~ The Email address of the User.
- **game\_username** ~ The username from an Asset Issuer's database to link to a User.
- **game\_uid** ~ The Unique Identifier from an Asset Issuer's database to link a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var game_email = "";
var game_username = "";
var game_uid = "";
sdk.dispatchExternalInvite(asset_issuer_uid, game_email, game_username, game_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# ConsumeExternalInvite

## SDK Type: Javascript/Node.js

**Description** Activates keys for a User.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **invite\_key** ~ The Invite Key sent to the User's email (or given to a user).

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var invite_key = "";
sdk.consumeExternalInvite(asset_issuer_uid, invite_key).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateUserAssetIssuerAccessGrant

## SDK Type: *Javascript/Node.js*

**Description** Grants access to inventory outside of a Game Session.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.createUserAssetIssuerAccessGrant(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# DeleteUserAssetIssuerAccessGrant

## SDK Type: Javascript/Node.js

**Description** Deletes access to inventory outside of a Game Session.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.deleteUserAssetIssuerAccessGrant(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# AssignItem

## SDK Type: Javascript/Node.js

**Description** Assigns a minted Asset to a User that has not been previously assigned.

### Required Inputs

- **asset\_uid** ~ An Asset that has been minted from an Asset Schema, but has not been assigned to a User.
- **recipient\_uid** ~ The User UID to receive an Asset.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_uid = "";
var recipient_uid = "";
sdk.assignItem(asset_uid, recipient_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# LoginUser

## SDK Type: Javascript/Node.js

**Description** Enables User log in.

### Required Inputs

- **user\_email** ~ A User's email address.
- **user\_password** ~ A User's password.

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var user_email = "";
var user_password = "";
sdk.loginUser(user_email, user_password).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UnlockUserInventory

## SDK Type: Javascript/Node.js

**Description** Unlocks a User's inventory during a Game Session if the Game Session is locked.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [GrantUserOwnership](#)
4. [CreateGameSession](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.unlockUserInventory(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UnlockUserInventoryOAuth

## SDK Type: Javascript/Node.js

**Description** Unlocks a User's inventory during a Game Session if the Game Session is locked.

### Required Inputs

- **oauth\_access\_token** ~ This is a Token that is obtained via the oAuth2 Implementation in Contact Systems.
- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [GrantUserOwnership](#)
4. [CreateGameSession](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var oauth_access_token = "";
var asset_issuer_uid = "";
var user_uid = "";
sdk.unlockUserInventoryOAuth(oauth_access_token, asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# LockUserInventory

## SDK Type: Javascript/Node.js

**Description** Locks a User's inventory during a Game Session if the Game Session is unlocked.

### Required Inputs

- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [GrantUserOwnership](#)
4. [CreateGameSession](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.lockUserInventory(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# LockUserInventoryOAuth

## SDK Type: Javascript/Node.js

**Description** Locks a User's inventory during a Game Session if the Game Session is unlocked.

### Required Inputs

- **oauth\_access\_token** ~ This is a Token that is obtained via the oAuth2 Implementation in Contact Systems.
- **asset\_issuer\_uid** ~ The UID of an Asset Issuer, which is defined as an entity that creates, issues, and manages digital assets.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [GrantUserOwnership](#)
4. [CreateGameSession](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var oauth_access_token = "";
var asset_issuer_uid = "";
var user_uid = "";
sdk.lockUserInventoryOAuth(oauth_access_token, asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# MintAssetSchemaForUser

## SDK Type: Javascript/Node.js

**Description** Enables an Asset Issuer to mint an Asset Schema into an Asset & then assign it to a User's inventory.

### Required Inputs

- **asset\_schema\_uid** ~ The UID of an Asset Schema, which is defined by the unique properties that make up a digital asset.
- **user\_uid** ~ The UID of a User.
- **json\_payload** ~ The JSON Schema Payload, which is required to mint an Asset.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var user_uid = "";
var json_payload = "";
sdk.mintAssetSchemaForUser(asset_schema_uid, user_uid, json_payload).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UpdateInventoryAsset

## SDK Type: Javascript/Node.js

**Description** This allows an Asset Issuer to Update an Asset within a User's inventory.

### Required Inputs

- **inventory\_uid** ~ *The UID of an Inventory Asset stored on a User's Account.*
- **json\_payload** ~ *The JSON Schema Payload, which is required to mint an Asset.*

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [MintAssetSchemaForUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var inventory_uid = "";
var json_payload = "";
sdk.updateInventoryAsset(inventory_uid, json_payload).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetUserInventoryAsset

## SDK Type: Javascript/Node.js

**Description** Gets Asset data for an Asset within a User's inventory.

### Required Inputs

- **asset\_uid** ~ *This is a Asset within a User's Inventory.*

### Notes

*Prerequisites are Listed below, if any are required.*

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_uid = "";
sdk.getUserInventoryAsset(asset_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetAssetIssuer

## SDK Type: Javascript/Node.js

**Description** Gets an Asset Issuer's data.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
sdk.getAssetIssuer(asset_issuer_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetAllAssetSchemas

## SDK Type: Javascript/Node.js

**Description** Gets all Asset Schemas created by an Asset Issuer.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
sdk.getAllAssetSchemas(asset_issuer_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetAssetSchema

## SDK Type: *Javascript/Node.js*

**Description** Gets a designated Asset Schema created by an Asset Issuer.

### Required Inputs

- **asset\_schema\_uid** ~ *The UID of an Asset Schema, which is defined by the unique properties that make up a digital asset.*

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **CreateAssetSchema**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
sdk.getAssetSchema(asset_schema_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateAssetSchema

## SDK Type: Javascript/Node.js

**Description** Creates an Asset Schema.

### Required Inputs

- **schema\_name** ~ An Asset Schema's name.
- **schema\_json** ~ A JSON Schema, which is required to create an Asset Schema <https://json-schema.org/>
- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **isactive** ~ Enables Asset Schemas to be minted; if "Inactive", the developer is unable to mint new Assets.
- **istradable** ~ Enables minted Assets to be traded between Users.
- **issellable** ~ Enables minted Assets to be sold.
- **isrevocable** ~ Enables an Asset Issuer to remove a minted Asset from a User's inventory.
- **isburnable** ~ Enables an Asset Issuer delete a minted Asset from a User's inventory.
- **tags** ~ List of Tags for a Associated Asset Schema Max is 5.
- **category** ~ Category ID for a contact systems category.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var schema_name = "";
var schema_json = "";
var asset_issuer_uid = "";
var isactive = false;
var istradable = false;
var issellable = false;
var isrevocable = false;
var isburnable = false;
var tags = "";
var category = "";
sdk.createAssetSchema(schema_name, schema_json, asset_issuer_uid, isactive, istradable, issellable,
isrevocable, isburnable, tags, category).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateAssetSchemaWithImage

## SDK Type: Javascript/Node.js

**Description** Creates an Asset Schema.

### Required Inputs

- **schema\_name** ~ An Asset Schema's name.
- **schema\_json** ~ A JSON Schema, which is required to create an Asset Schema <https://json-schema.org/>
- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **isactive** ~ Enables Asset Schemas to be minted; if "Inactive", the developer is unable to mint new Assets.
- **istradable** ~ Enables minted Assets to be traded between Users.
- **issellable** ~ Enables minted Assets to be sold.
- **isrevocable** ~ Enables an Asset Issuer to remove a minted Asset from a User's inventory.
- **isburnable** ~ Enables an Asset Issuer delete a minted Asset from a User's inventory.
- **tags** ~ List of Tags for a Associated Asset Schema Max is 5.
- **category** ~ Category ID for a contact systems category.
- **image\_file** ~ Directory location of an image file.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var schema_name = "";
var schema_json = "";
var asset_issuer_uid = "";
var isactive = false;
var istradable = false;
var issellable = false;
var isrevocable = false;
var isburnable = false;
var tags = "";
var category = "";
var image_file = "";
sdk.createAssetSchemaWithImage(schema_name, schema_json, asset_issuer_uid, isactive, istradable,
issellable, isrevocable, isburnable, tags, category, image_file).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetUserFromGameSession

## SDK Type: Javascript/Node.js

**Description** Gets a User's data from a Game Session.

### Required Inputs

- **user\_session** ~ *The User Session passed from the Desktop Login App.*

### Notes

*Prerequisites are Listed below, if any are required.*

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **CreateGameSession**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var user_session = "";
sdk.getUserFromGameSession(user_session).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateGameSession

## SDK Type: Javascript/Node.js

**Description** Creates a Game Session for a User.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.createGameSession(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateGameSessionOAuth

## SDK Type: Javascript/Node.js

**Description** Creates a Game Session for a User.

### Required Inputs

- **oauth\_access\_token** ~ This is a Token that is obtained via the oAuth2 Implementation in Contact Systems.
- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **LinkAssetIssuerAccountToUser**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var oauth_access_token = "";
var asset_issuer_uid = "";
var user_uid = "";
sdk.createGameSessionOAuth(oauth_access_token, asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateLockedGameSession

## SDK Type: Javascript/Node.js

**Description** Creates a Game Session with the inventory locked by an Asset Issuer for a User.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.createLockedGameSession(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateLockedGameSessionOAuth

## SDK Type: Javascript/Node.js

**Description** Creates a Game Session with the inventory locked by an Asset Issuer for a User.

### Required Inputs

- **oauth\_access\_token** ~ This is a Token that is obtained via the oAuth2 Implementation in Contact Systems.
- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **LinkAssetIssuerAccountToUser**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var oauth_access_token = "";
var asset_issuer_uid = "";
var user_uid = "";
sdk.createLockedGameSessionOAuth(oauth_access_token, asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# EndGameSession

## SDK Type: *Javascript/Node.js*

**Description** Ends a Game Session for a User.

---

### Required Inputs

- **user\_uid** ~ *The UID of a User.*

### Notes

*Prerequisites are Listed below, if any are required.*

1. **CreateOrganization**
  2. **CreateAssetIssuer**
  3. **CreateGameSession**
- 

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var user_uid = "";
sdk.endGameSession(user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# EndGameSessionOAuth

## SDK Type: Javascript/Node.js

**Description** Ends a Game Session for a User.

### Required Inputs

- **oauth\_access\_token** ~ This is a Token that is obtained via the oAuth2 Implementation in Contact Systems.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateGameSession](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var oauth_access_token = "";
var user_uid = "";
sdk.endGameSessionOAuth(oauth_access_token, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# LinkAssetIssuerAccountToUser

## SDK Type: Javascript/Node.js

**Description** Links an Asset Issuer's account to a CONTACT Systems User account.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **user\_uid** ~ The UID of a User.
- **game\_username** ~ This is the Username from the Asset Issuer's Database to Link to a User within CONTACT Systems.
- **game\_uid** ~ This is the Unique Identifier from the Asset Issuer's Database to Link a User within CONTACT Systems.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
var game_username = "";
var game_uid = "";
sdk.linkAssetIssuerAccountToUser(asset_issuer_uid, user_uid, game_username, game_uid).then((data) =>
{
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UnlinkAssetIssuerAccountFromUser

## SDK Type: Javascript/Node.js

**Description** Unlinks an Asset Issuer's account to a CONTACT Systems User account.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **user\_uid** ~ The UID of a User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var user_uid = "";
sdk.unlinkAssetIssuerAccountFromUser(asset_issuer_uid, user_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UpdateAssetIssuer

## SDK Type: Javascript/Node.js

**Description** Updates an Asset Issuer.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **issuer\_name** ~ An Asset Issuer's name.
- **issuer\_industry** ~ Name of the Industry the Asset Issuer is apart of.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var issuer_name = "";
var issuer_industry = "";
sdk.updateAssetIssuer(asset_issuer_uid, issuer_name, issuer_industry).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateAssetIssuer

## SDK Type: Javascript/Node.js

**Description** Creates an Asset Issuer.

### Required Inputs

- **organization\_uid** ~ This is the UID of a Organization with-in CONTACT System's CoreAPI
- **issuer\_name** ~ An Asset Issuer's name.
- **issuer\_industry** ~ Name of the Industry the Asset Issuer is apart of.

### Notes

Prerequisites are Listed below, if any are required.

#### 1. CreateOrganization

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var organization_uid = "";
var issuer_name = "";
var issuer_industry = "";
sdk.createAssetIssuer(organization_uid, issuer_name, issuer_industry).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UpdateAssetIssuerWithImage

## SDK Type: Javascript/Node.js

**Description** Updates an Asset Issuer with an image.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **issuer\_name** ~ An Asset Issuer's name.
- **issuer\_industry** ~ Name of the Industry the Asset Issuer is apart of.
- **image\_file** ~ Directory location of an image file.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var issuer_name = "";
var issuer_industry = "";
var image_file = "";
sdk.updateAssetIssuerWithImage(asset_issuer_uid, issuer_name, issuer_industry,
image_file).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateAssetIssuerWithImage

## SDK Type: Javascript/Node.js

**Description** Creates an Asset Issuer with an image.

### Required Inputs

- **organization\_uid** ~ This is the UID of a Organization with-in CONTACT System's CoreAPI
- **issuer\_name** ~ An Asset Issuer's name.
- **issuer\_industry** ~ Name of the Industry the Asset Issuer is apart of.
- **image\_file** ~ Directory location of an image file.

### Notes

Prerequisites are Listed below, if any are required.

#### 1. CreateOrganization

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var organization_uid = "";
var issuer_name = "";
var issuer_industry = "";
var image_file = "";
sdk.createAssetIssuerWithImage(organization_uid, issuer_name, issuer_industry,
image_file).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UpdateAssetSchemaWithImage

## SDK Type: Javascript/Node.js

**Description** Updates an existing Asset Issuer.

### Required Inputs

- **asset\_schema\_uid** ~ This is the UID of a Asset Schema that was Created by the Asset Issuer making the Request.
- **schema\_name** ~ An Asset Schema's name.
- **schema\_json** ~ A JSON Schema, which is required to create an Asset Schema <https://json-schema.org/>
- **issellable** ~ Enables minted Assets to be sold.
- **istradable** ~ Enables minted Assets to be traded between Users.
- **isrevocable** ~ Enables an Asset Issuer to remove a minted Asset from a User's inventory.
- **isburnable** ~ Enables an Asset Issuer delete a minted Asset from a User's inventory.
- **isactive** ~ Enables Asset Schemas to be minted; if "Inactive", the developer is unable to mint new Assets.
- **tags** ~ List of Tags for a Associated Asset Schema Max is 5.
- **category** ~ Category ID for a contact systems category.
- **image\_file** ~ Directory location of an image file.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **CreateAssetSchema**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var schema_name = "";
var schema_json = "";
var issellable = false;
var istradable = false;
var isrevocable = false;
var isburnable = false;
var isactive = false;
var tags = "";
var category = "";
var image_file = "";
sdk.updateAssetSchemaWithImage(asset_schema_uid, schema_name, schema_json, issellable, istradable,
isrevocable, isburnable, isactive, tags, category, image_file).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UpdateAssetSchema

## SDK Type: Javascript/Node.js

**Description** Updates an existing Asset Issuer.

### Required Inputs

- **asset\_schema\_uid** ~ This is the UID of a Asset Schema that was Created by the Asset Issuer making the Request.
- **schema\_name** ~ An Asset Schema's name.
- **schema\_json** ~ A JSON Schema, which is required to create an Asset Schema <https://json-schema.org/>
- **issellable** ~ Enables minted Assets to be sold.
- **istradable** ~ Enables minted Assets to be traded between Users.
- **isrevocable** ~ Enables an Asset Issuer to remove a minted Asset from a User's inventory.
- **isburnable** ~ Enables an Asset Issuer delete a minted Asset from a User's inventory.
- **isactive** ~ Enables Asset Schemas to be minted; if "Inactive", the developer is unable to mint new Assets.
- **tags** ~ List of Tags for a Associated Asset Schema Max is 5.
- **category** ~ Category ID for a contact systems category.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [CreateAssetSchema](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
var schema_name = "";
var schema_json = "";
var issellable = false;
var istradable = false;
var isrevocable = false;
var isburnable = false;
var isactive = false;
var tags = "";
var category = "";
sdk.updateAssetSchema(asset_schema_uid, schema_name, schema_json, issellable, istradable,
isrevocable, isburnable, isactive, tags, category).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# RevokeInventoryAsset

## SDK Type: Javascript/Node.js

**Description** Revokes an Asset within a user's inventory.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.
- **asset\_uid** ~ This is the UID of a Asset within a User's Inventory within CONTACT Systems.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **LinkAssetIssuerAccountToUser**
4. **MintAssetSchemaForUser**
5. **CreateGameSession**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var asset_uid = "";
sdk.revokeInventoryAsset(asset_issuer_uid, asset_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# UseInventoryAsset

## SDK Type: Javascript/Node.js

**Description** Uses an Asset within a User's inventory.

### Required Inputs

- **asset\_uid** ~ *This is a Asset within a User's Inventory.*

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)
4. [MintAssetSchemaForUser](#)
5. [CreateGameSession](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_uid = "";
sdk.useInventoryAsset(asset_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# InviteUserFromAssetIssuer

## SDK Type: Javascript/Node.js

**Description** Invites a User from an Asset Issuer's User System into the CØNTACT System's User System.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CØNTACT Systems API.
- **game\_username** ~ This is the Username provided by the Asset Issuer to create a CØNTACT Systems Account by Linking their Asset Issuer account to CØNTACT.
- **game\_uid** ~ This is the Asset Issuer's Unique Identifier to identify the user in their system.
- **game\_email** ~ This is the Email address of the User.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
var game_username = "";
var game_uid = "";
var game_email = "";
sdk.inviteUserFromAssetIssuer(asset_issuer_uid, game_username, game_uid, game_email).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# ListAssetIssuerAccounts

## SDK Type: Javascript/Node.js

**Description** Lists all linked accounts for an Asset Issuer.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)
3. [LinkAssetIssuerAccountToUser](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
sdk.listAssetIssuerAccounts(asset_issuer_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# DisableAssetSchema

## SDK Type: Javascript/Node.js

**Description** Disables an enabled Asset Schema.

### Required Inputs

- **asset\_schema\_uid** ~ *This is the UID of a Asset Schema that was Created by the Asset Issuer making the Request.*

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **CreateAssetSchema**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
sdk.disableAssetSchema(asset_schema_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# EnableAssetSchema

## SDK Type: Javascript/Node.js

**Description** Enables a disabled Asset Schema.

### Required Inputs

- **asset\_schema\_uid** ~ *This is the UID of a Asset Schema that was Created by the Asset Issuer making the Request.*

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**
3. **CreateAssetSchema**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_schema_uid = "";
sdk.enableAssetSchema(asset_schema_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# DisableAssetIssuer

## SDK Type: Javascript/Node.js

**Description** Disables a Enabled Asset Issuer

---

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
  2. [CreateAssetIssuer](#)
- 

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
sdk.disableAssetIssuer(asset_issuer_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# EnableAssetIssuer

## SDK Type: Javascript/Node.js

**Description** Enables a disabled Asset Issuer.

### Required Inputs

- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)
2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var asset_issuer_uid = "";
sdk.enableAssetIssuer(asset_issuer_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# CreateOauth2Client

## SDK Type: Javascript/Node.js

**Description** Allows a Developer to create a oAuth 2 Client.

### Required Inputs

- **domain** ~ This is the Domain URL for the Redirection System.
- **description** ~ This allows you to Set what this oAuth 2 Client is for.
- **asset\_issuer\_uid** ~ This is the UID of a Asset Issuer which can be a Game, Application or Service that interfaces with CONTACT Systems API.

### Notes

Prerequisites are Listed below, if any are required.

1. **CreateOrganization**
2. **CreateAssetIssuer**

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var domain = "";
var description = "";
var asset_issuer_uid = "";
sdk.createOauth2Client(domain, description, asset_issuer_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# GetUserDataOAuth

## SDK Type: Javascript/Node.js

**Description** Gets user's base data based on OAuth

---

### Required Inputs

- **oauth\_access\_token** ~ *This is a Token that is obtained via the oAuth2 Implementation in Contact Systems.*

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)

2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var oauth_access_token = "";
sdk.getUserDataOAuth(oauth_access_token).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# ListCategories

## SDK Type: Javascript/Node.js

**Description** Lists all Categories on the Contact Platform.

### Required Inputs

### Notes

*Prerequisites are Listed below, if any are required.*

#### 1. None

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);

sdk.listCategories().then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```



# ListAssetIssuers

## SDK Type: Javascript/Node.js

**Description** Lists all Asset Issuers for a Organization

### Required Inputs

- **organization\_uid** ~ This is the UID of a Organization with-in CONTACT System's CoreAPI

### Notes

Prerequisites are Listed below, if any are required.

1. [CreateOrganization](#)

2. [CreateAssetIssuer](#)

```
var access = "00000000-0000-0000-0000-000000000000", secret = "00000000-0000-0000-0000-000000000000";
var contact = require('metropolis-node'), sdk = new contact(access, secret);
var organization_uid = "";
sdk.listAssetIssuers(organization_uid).then((data) => {
  console.log(data);
}).catch((err) => {
  console.log(err);
});
```