Import linked content

January 24, 2023 • Jan Cerman • 5 min read • JavaScript

Your content will often contain references to other pieces of imported content. A content item can use assets or point to other content items using linked items elements or rich text elements.

This tutorial will walk through how to import these references to Kontent.ai. For the basics of using Management API, first see Importing to Kontent.ai.

How to think about links between your content

To avoid having to import objects in a specific order, use external IDs to reference content that's not yet imported. This solves problems with circular dependencies and let's you reference non-existent content.

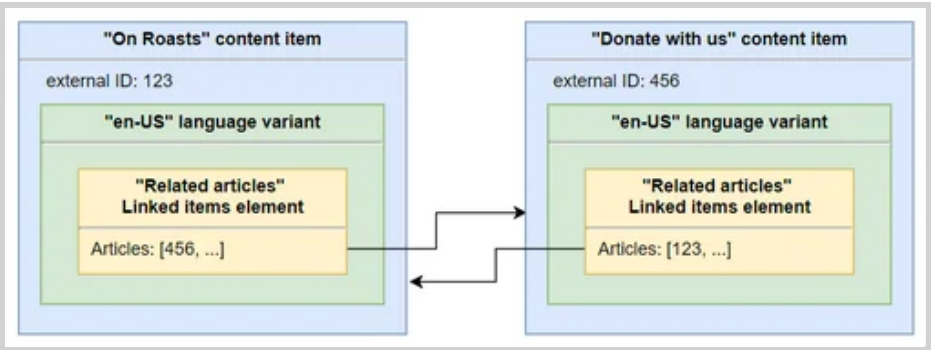
Here is how it works:

1. Define external IDs for all content items and assets you want to import in advance.
2. When referencing another content item or asset, use its external ID.
3. Import your content using the upsert methods with an external ID. The system will resolve all references.

This way, you can import your content in any order and run the import process repeatedly to keep your project up to date. In the example below, you will import two content items that reference each other in their Linked items elements.

Example scenario

Say you want to import two related content items: *Donate with us* and *On Roasts* articles. Each content item references the other in the *Related articles* Linked items element. The result will have the following structure:



1. Define external IDs

External IDs are string-based identifiers of items and assets defined by you. You can define new IDs or reuse IDs from the original storage system you are importing content from.

It's up to you to ensure no two objects have the same external ID. See more details on using external IDs for imported content. For large projects, consider using GUIDs.

To keep things simple here, use `123` and `456` for your two articles.

2. Use external IDs to reference items

When defining the Linked items elements, use external IDs to reference the other content item:

JSON

```
"related_articles": [
  {
    "external_id": "456"
  }
]
```

Rich text links

See how to import rich text to learn how to link items in rich text elements.

3. Import content

To create a content item, send a PUT request to the `/items/external-id/<YOUR_ITEM_EXTERNAL_ID>` endpoint.

In the body of the request, specify the item's name and content type.

See more details on upserting content items.

Best practice: Upsert by external ID

You can use a simple POST to /items request to add the content item. But using an UPSERT operation and defining an external ID for your item has advantages and makes the import process much smoother:

- You can run the same request repeatedly. If the item doesn't exist, it will be created. If it does, it will be updated.
- You can reference or link to your item, even if it hasn't been imported yet (and has no internal ID or codename). You might have other content items that reference this one in Rich text or Linked items elements. But if you are using external IDs you don't need to worry about the order in which the content items are imported.

JavaScript

```
// Tip: Find more about JS/TS SDKs at https://kontent.ai/learn/javascript
// Using ES6 syntax
import { ManagementClient } from '@kontent-ai/management-sdk';

const client = new ManagementClient({
  environmentId: '<YOUR_ENVIRONMENT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

const response = await client.upsertContentItem()
  .byItemExternalId('123')
  .withData(
    {
      name: 'On Roasts',
      type: 'article'
    }
  )
  .toPromise();
```

Import content by upserting a language variant.

Send a PUT request to the endpoint specifying the language variant you want to insert or update. In the body of the request, specify the values of individual content elements.

JavaScript

```
// Tip: Find more about JS/TS SDKs at https://kontent.ai/learn/javascript
// Using ES6 syntax
import { ManagementClient } from '@kontent-ai/management-sdk';

const client = new ManagementClient({
  environmentId: '<YOUR_ENVIRONMENT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

const response = await client.upsertLanguageVariant()
  .byItemExternalId('123')
  .byLanguageCodename('en-US')
  .withElements([
    {
      element: {
        codename: 'title'
      },
      value: 'On Roasts'
    },
    {
      element: {
        codename: 'related_articles'
      },
```

```
      value: [
        {
          "external_id":"456"
        }
      ]
    }
  ])
  .toPromise();
```

Notice that you are referencing the *Donate with us* item even though you haven't imported it yet.

The reference is not visible inside the Kontent.ai UI, but it still exists. It will resolve itself once you import the second content item.

Second content item

Repeat the same process with the *Donate with us* article. Start by creating the content item:

JavaScript

```
// Tip: Find more about JS/TS SDKs at https://kontent.ai/learn/javascript
// Using ES6 syntax
import { ManagementClient } from '@kontent-ai/management-sdk';

```

```
const client = new ManagementClient({
  environmentId: '<YOUR_ENVIRONMENT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

const response = await client.upsertContentItem()
  .byItemExternalId('456')
  .withData(
    {
      name: 'Donate with us',
      type: 'article'
    }
  )
  .toPromise();
```

This resolves the reference in the *On Roasts* item.

Lastly, import the content of the *Donate with us* item by upserting its language variant:

JavaScript

```
// Tip: Find more about JS/TS SDKs at https://kontent.ai/learn/javascript
// Using ES6 syntax
import { ManagementClient } from '@kontent-ai/management-sdk';

const client = new ManagementClient({
  environmentId: '<YOUR_ENVIRONMENT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

const response = await client.upsertLanguageVariant()
  .byItemExternalId('456')
  .byLanguageCodename('en-US')
  .withElements([
    {
      element: {
        codename: 'title'
      },
      value: 'Donate with us'
    },
    {
      element: {
        codename: 'related_articles'
      },
      value: [
        {
          "external_id":"123"
        }
      ]
    }

```

```
30    ])
31    .toPromise();
```

Both references are now resolved. To verify, you can view the imported content items in Kontent.ai:

In Kontent.ai, select ***Content & assets*** *to view the imported content items.*

Validate imported content

After your import process is finished, we recommend that you validate your project's content for inconsistencies. For example, if you forget to add the second item, the validation would report that you're referencing a nonexistent object. Project validation can also reveal other issues like references to missing assets or empty required elements.

What's next?

You've imported two content items that reference one another using their external IDs.

- Dive deep into the details of Management API.
- Learn about working with assets via API.
- Learn about importing rich text.