Import assets

June 28, 2022 • Jan Cerman • 6 min read • JavaScript

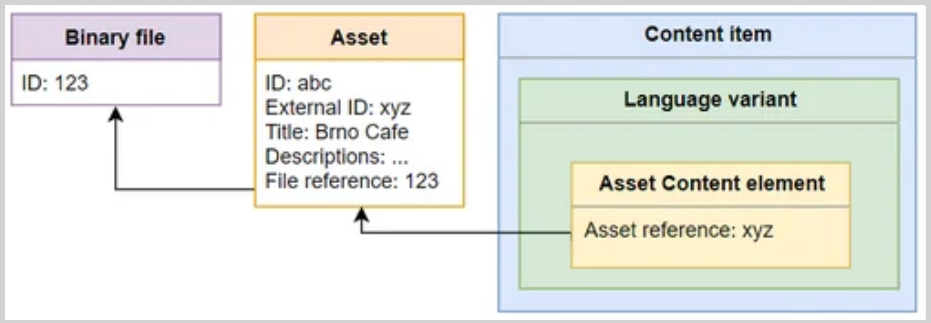
Let's learn how to import a single asset to Kontent.ai and use it in a content item. For the basics of using the Management API, first see Importing to Kontent.ai.

Asset import in a nutshell

Importing assets using the Management API is a 3-step process.

1. Upload a file to your Kontent.ai project.
2. Create a new asset with a reference to the uploaded file.
3. Link to the asset from a content item's language variant.

The result has the following structure:



1. Upload files

To upload a file via API, send a POST request to the `<YOUR_PROJECT_ID>/files/<YOUR_FILE_NAME>` endpoint with your project ID and your file name filled in. For example, `975bf280-fd91-488c-994c-2f04416e5ee3/files/my%20file%232.png`.

URL-encode reserved characters

If the name of your file contains reserved characters, you need to URL encode the file name first. For instance, use `my%20file%232.png` instead of `my file#2.png`. The file will then show as `my file#2.png` in the Kontent.ai UI.

- Set the `Content-type` header to the MIME type matching your file. For example, for a jpg image set the header to `image/jpeg`.
- Set the `Content-length` header to a number matching the size of the file in bytes. For example, if the image is 12 kB, set the header to `12288`.

Don't forget to set your Management API key in the request's authorization header (or the `ManagementClient` definition when using an SDK).

You can only upload files from local storage. The Management API doesn't support uploading files from a URL.

JavaScript

```
// Using ES6 syntax
// This approach works when using Node.js.
import { ManagementClient } from '@kontent-ai/management-sdk';
import { readFileSync } from 'fs'

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

const data = readFileSync('./<YOUR_PATH>/brno-cafe-1080px.jpg');

const response = await client.uploadBinaryFile()
  .withData({
    binaryData: data,
    contentLength: data.byteLength,
    contentType: 'image/jpeg',
    filename: 'brno-cafe-1080px.jpg'
  })
  .toPromise();
```

This returns a `file_reference` object.

JSON

```
{
    "id": "8660e19c-7bbd-48a3-bb51-721934c7756c",
    "type": "internal"
}
```

You'll use the `file_reference` object to create the actual asset in the next step.

2. Create assets

To create an asset using the uploaded file, it's best to perform an upsert operation. This way, you can send the request repeatedly to update your asset.

This means sending a PUT request to the `<YOUR_PROJECT_ID>/assets/external-id/<ASSET_EXTERNAL_ID>` endpoint with your project ID and an external ID of your choice for the asset filled in.

Best practice: Referencing by external ID

Make it easier to reference your asset from content items by specifying an external ID. External IDs can point to non-existent (not yet imported) content. This way, you can import your assets and content items in any order and not worry about dependencies.

See referencing by external ID for more details.

In the body of the request, specify these properties:

- `file_reference` – use the file reference you obtained in the previous step to link the asset with your file.
- (Optional) `title` – specify a title for the asset (displayed in the administration interface).
- `descriptions` – specify an asset description for each language in your project.
  - `language` – can be specified by its `codename` or internal `id`.
  - `description` – string with a description of the asset.

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({
  projectId: '<YOUR_PROJECT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

// Uses the file reference object obtained in step 1
const response = await client.upsertAsset()
  .byAssetExternalId('brno-cafe-image')
  .withData(
    {
      file_reference: {
        id: '8660e19c-7bbd-48a3-bb51-721934c7756c',
        type: 'internal'
      },
      title: "Brno Cafe",
      descriptions: [
        {
          language: {
            codename: 'en-US'
          },
          description: 'Cafe in Brno'
        },
        {
          language: {
            codename: 'es-ES'
          },
          description: 'Café en Brno'
        }
      ]
    }
  )
  .toPromise();
```

See our API reference for more details on upserting assets by external id.

The response will include an asset object:

JSON

```
{
  "id": "8d0baeb7-1165-5f13-b72e-b23fc39fc549",
  "file_name": "brno-cafe-1080px.jpg",
  "title": "Brno Cafe",
  "size": 481939,
  "image_width": 1920,
  "image_height": 1440,
  "type": "image/jpeg",
  "file_reference": {
    "id": "8660e19c-7bbd-48a3-bb51-721934c7756c",
    "type": "internal"
  },
  "descriptions": [
    {
      "language": {
        "id": "00000000-0000-0000-0000-000000000000"
      },
      "description": "Cafe in Brno"
    },
    {
      "language": {
        "id": "d1f95fde-af02-b3b5-bd9e-f232311ccab8"
      },
      "description": "Café en Brno"
    }
  ],
  "external_id": "brno-cafe-image",
  "last_modified": "2021-04-10T09:44:39.0111761Z"
}
```

To verify the creation of the asset, you can view it in Kontent.ai:

Go to ***Content & assets*** *>* ***Assets*** *to view and edit your files.*

3. Use assets in content items

The content of an item is stored in a specific language variant. Within the language variant, you can use the imported asset in asset and rich text elements. The asset might need to meet the limitations set for the item's content type.

Use assets in Asset elements

To use the imported asset in an asset element, you need to upsert a language variant by sending a PUT request to an endpoint specifying the content item (for example, `/items/external-id/ext-cafe-brno`) and the language of the variant (for example, `/variants/codename/en-US`). (This is the same item you worked with in Importing to Kontent.ai.)

The request body must contain a single `elements` object. In it, you specify the content elements you want to update – in this case, only the `photo` asset element. The asset element takes an array of Reference objects, each containing either the internal ID or external ID of the asset.

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({
  projectId: '<YOUR_PROJECT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

```

```
const response = await client.upsertLanguageVariant()
  .byItemExternalId('ext-cafe-brno')
  .byLanguageCodename('en-US')
  .withElements([
    {
      element: {
        codename: 'photo'
      },
      value: [
        {
          external_id: "brno-cafe-image"
        }
      ]
    }
  ])
  .toPromise();
```

The language variant of the item was updated and the *Photo* element now contains the imported asset. You can verify this by opening the item in Kontent.ai:

Use assets in rich text elements

You can also use assets as inline images in rich text elements.

For this example, we created a new item of the *Article* content type (which contains a Rich text element) with the external ID `new-cafes`.

To add your asset, you need to upsert a language variant by sending a PUT request to the `/items/external-id/new-cafes/variants/codename/en-US` endpoint, specifying the content item and the language of the variant.

The request body must contain a single `elements` object. In it, you specify the content elements you want to update – in this case, only the `body_copy` rich text element. Use the `<figure>` HTML element to add an image using its external ID.

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({
  projectId: '<YOUR_PROJECT_ID>',
  apiKey: '<YOUR_API_KEY>'
});

const response = await client.upsertLanguageVariant()
  .byItemExternalId('new-cafes')
  .byLanguageCodename('en-US')
  .withElements([
    {
      element: {
        codename: 'body_copy'
      },
      value: "<p>...</p> <figure data-asset-external-id=\"brno-cafe-image\"></figure>"
    }
  ])
  .toPromise();
```

See the rich text model in our API Reference for more examples of links inside rich text elements.

What's next?

You uploaded a file to Kontent.ai, created an asset for it, and used the asset in a content item.

- For more information on working with assets and other content using the Management API, see our API Reference.
- Have a look at our Management SDKs for .NET and JavaScript.
- See how to import rich text and linked content.