Import content items

June 28, 2022 • Tomas Nosek • 6 min read • JavaScript

This tutorial will guide you through importing a single content item by using the Management API. The Management API is a secure REST API that provides read/write access to your Kontent.ai projects.

In a nutshell, to import content, you define a content type for the imported content if it's not created already. Then, you'll create a new content item and add content to its variants.

If you haven't already, enable the Management API in your project before starting.

> **Importing taxonomies, snippets, and content groups**
>
> Import your content model before importing content items. This tutorial continues without taxonomies, snippets, and content groups for the sake of simplicity. In general, you'd add a reference to them when importing the content items.

1. Define a content type for the content item

Before importing your content, you need to define a content type for the items you want to import. The content type in this example will be named *Cafe* (with the codename and external ID both as *cafe*) and contain a couple of elements.

To add a content type, send a POST request with these properties:

- The display name of the new content type.
- (Optional) The codename of the content type. If you don't specify the codename, it will be generated from the content type's display name. However, you can change the code name later.
- (Optional) The external ID, which can be used as a unique identifier to retrieve content from a different system. See more about how to use different identifiers.
- A set of the elements you want to have in your content type.

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

JavaScript

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

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

```
7
8  const response = await client.addContentType()
9    .withData(
10     {
11       codename: "cafe",
12       name: "Cafe",
13       external_id: "cafe",
14       elements: [
15         {
16           name: "Price per unit",
17           type: "number",
18           codename: "price_per_unit"
19         },
20         {
21           guidelines: "<h2>Keep Guidelines where the creative process happens.
22 </h2>\n<p>These are sample guidelines that you can place for the whole content item.
23 It’s a place where you can include your content brief, voice and tone recommendations or
   the URL to a wireframe, so the author will have all the relevant instructions at hand
   before writing a single line.</p>\n<p>Besides overview guidelines, you can include
   instructions for each particular content element, as you will see below.</p>",
           type: ElementModels.ElementType.guidelines,
           codename: "n2f836bce_e062_b2cd_5265_f5c3be3aa6f5"
         },
         {
           name: "Street",
24         type: ElementModels.ElementType.text,
25         codename: "street"
26       },
27       {
28         name: "City",
29         type: ElementModels.ElementType.text,
30         codename: "city"
31       },
32       {
33         name: "Country",
34         type: ElementModels.ElementType.text,
35         codename: "country"
36       },
37       {
38         name: "State",
39         type: ElementModels.ElementType.text,
40         codename: "state"
41       },
42       {
43         name: "ZIP Code",
44         type: ElementModels.ElementType.text,
45         codename: "zip_code"
46       },
47       {
48         name: "Phone",
```

```
          type: ElementModels.ElementType.text,
          codename: "phone"
        },
        {
          name: "Email",
          type: ElementModels.ElementType.text,
          codename: "email"
        },
        {
          name: "Photo",
          type: ElementModels.ElementType.asset,
          codename: "photo"
        }
      ]
    }
  )
  .toPromise();
```

After making the request, you'll get a JSON response like this:

JSON


```
{
  "id": "f15799a1-dff0-5216-a014-e963f9ea6bbc",
  "codename": "cafe",
  "last_modified": "2019-10-08T12:47:27.4261844Z",
  "external_id": "cafe",
  "name": "Cafe",
  "content_groups": [],
  "elements": [
    {
      "name": "Price per unit",
      "guidelines": null,
      "is_required": false,
      "type": "number",
      "id": "965cf18c-998a-456b-bee3-91defdf39782",
      "codename": "price_per_unit"
    },
    {
      "guidelines": "<h2>Keep Guidelines where the creative process happens.
</h2>\n<p>These are sample guidelines that you can place for the whole content item.
It’s a place where you can include your content brief, voice and tone recommendations or
the URL to a wireframe, so the author will have all the relevant instructions at hand
before writing a single line.</p>\n<p>Besides overview guidelines, you can include
instructions for each particular content element, as you will see below.</p>",
      "type": "guidelines",
      "id": "8816e7d0-ca5e-4cfa-88b4-c1944862ff9f",
      "codename": "n8e317a38_e3bc_4d98_a63d_f1e336c5a9e6"
    },
    {
      "maximum_text_length": null,
```

```
      "name": "Street",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "2b7b256c-73cc-4be0-b749-4c410209df02",
      "codename": "street"
    },
    {
      "maximum_text_length": null,
      "name": "City",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "fa17f73a-833c-4cf2-aa7b-3b5c4edad6b3",
      "codename": "city"
    },
    {
      "maximum_text_length": null,
      "name": "Country",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "2b36b05e-5f4f-4630-b502-154f5b3b90c3",
      "codename": "country"
    },
    {
      "maximum_text_length": null,
      "name": "State",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "3638f2cd-3fea-4a4e-a6b6-bdd39f3b9f66",
      "codename": "state"
    },
    {
      "maximum_text_length": null,
      "name": "ZIP Code",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "35ad677a-cfe5-4573-814d-5895e3de8396",
      "codename": "zip_code"
    },
    {
      "maximum_text_length": null,
      "name": "Phone",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "4ec54355-4177-4b29-9a16-b25a7c8fba26",
```

```
      "codename": "phone"
    },
    {
      "maximum_text_length": null,
      "name": "Email",
      "guidelines": null,
      "is_required": false,
      "type": "text",
      "id": "9bc4b679-4353-40c7-bcfd-145110ced7d9",
      "codename": "email"
    },
    {
      "asset_count_limit": null,
      "maximum_file_size": null,
      "allowed_file_types": "any",
      "image_width_limit": null,
      "image_height_limit": null,
      "name": "Photo",
      "guidelines": null,
      "is_required": false,
      "type": "asset",
      "id": "1693ecd3-fb15-4d56-975a-4b8dbdedf65b",
      "codename": "photo"
    }
  ]
}
```


2. Import content to a content item

Importing a content item consists of two tasks. That's because, technically, each content item is a wrapper that contains language variants. Variants then hold the content. This enables content to be localized but the same approach applies to projects with one language as well.

What content items consist of

2a. Create a content item

To add a content item, send a PUT request with these properties:

- The name, which is a string representing the display name of the new content item.
- The codename of the content type, which is used for the content item.

As importing a lot of content can become confusing, handling content from your original system's point of view is typically easier. Use external IDs to identify your content when doing a migration.

With external IDs, you can also reference items that haven't been imported yet. That's when content items link each other in rich text or linked items elements. You don't need to worry about the order in which the content items are imported.

However, if you don't want to use external IDs, you can use other identifiers as well.

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_MANAGEMENT_API_KEY>'
});

const response = await client.upsertContentItem()
  .byItemExternalId('ext-cafe-brno')
  .withData(
    {
      name: 'Brno',
      type: 'cafe'
    }
  )
  .toPromise();
```

Why PUT is better than POST

Besides the PUT request from the example, you can also use a POST request. However, using a PUT operation has its advantages and makes the import process much smoother. With PUT requests, you can:

- Run the same request repeatedly – If the item doesn't exist, it will be created. If it does, it will be updated.
- Reference your items with external IDs

You can find out more about both topics in the API reference on upserting content items.

After sending the request, Kontent.ai generates the internal ID and codename for the content item and includes it in the response. Also, the content item is assigned to the default collection.

JSON

```
{
  "id": "8ceeb2d8-9676-48ae-887d-47ccb0f54a79",
  "name": "Brno",
  "codename": "brno",
  "type": {
    "id": "fe41ae5a-5fe2-420a-8560-f7d6d3533dc2"
  },
  "collection": {
    "id": "00000000-0000-0000-0000-000000000000"
  },
  "sitemap_locations": [],
  "external_id": "ext-cafe-brno",
  "last_modified": "2017-11-09T16:51:09.041611Z"
}
```

You've just created an empty content item.

To verify that the content item has been added, you have two options:

- Retrieve the content item using the Management API.
- Check your content list in Kontent.ai.

The created item will show up as Not translated *inside your project.*

So far, you have only specified the general attributes of the content item – its name, content type, and external ID. Now, let's add the actual (localized) content.

2b. Add (localized) content

The actual content is stored in the individual content elements in each variant. To add content there, send a PUT request and specify the following:

- The content item you're importing content to – ideally with its external ID. For example, *ext-cafe-brno*.
- The language's code name – available in ⚙ **Project settings** > **Localization**. For example, *en-US*.

- Elements with content that you want to import. Omitted elements will:
  - Remain empty when creating new content
  - Remain unchanged when updating already existing content
  - Generate automatically if they're *URL slugs*

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_MANAGEMENT_API_KEY>'
});

const response = await client.upsertLanguageVariant()
  .byItemExternalId('ext-cafe-brno')
  .byLanguageCodename('en-US')
  .withData((builder) => [
    builder.textElement({
      element: {
        codename: 'street'
      },
      value: 'Nove Sady 25'
    }),
    builder.textElement({
      element: {
        codename: 'city'
      },
      value: 'Brno'
    }),
    builder.textElement({
      element: {
        codename: 'country'
      },
      value: 'Czech Republic'
    }),
```

```
    builder.textElement({
      element: {
        codename: 'state'
      },
      value: 'Jihomoravsky kraj'
    }),
    builder.textElement({
      element: {
        codename: 'zip_code'
      },
      value: '60200'
    }),
    builder.textElement({
      element: {
        codename: 'phone'
      },
      value: '+420555555555'
    }),
    builder.textElement({
      element: {
        codename: 'email'
      },
      value: 'brnocafe@kontent.ai'
    })
  ])
  .toPromise();
```

The response will include the updated item variant.

JSON

```
{
    "elements": [
        {
            "element": {
                "id": "866afdba-d334-f01a-1d52-a9ca3f57cb4b"
            },
            "value": "Nove Sady 25"
        },
        {
            "element": {
                "id": "339e6d4f-67c1-5f5e-6921-3b374eb96f5b"
            },
            "value": "Brno"
        },
        {
            "element": {
                "id": "7531a08f-e148-8cc0-9d2d-155215502e08"
            },
            "value": "Czech Republic"
        },
```

```
        {
            "element": {
                "id": "a015b689-cad3-1ac9-04b4-73697525752d"
            },
            "value": "Jihomoravsky kraj"
        },
        {
            "element": {
                "id": "bb158ac2-41e1-5a7d-0826-bb8bf6744f0e"
            },
            "value": "60200"
        },
        {
            "element": {
                "id": "1c71bc62-4b62-f307-37ef-0823776f8f73"
            },
            "value": "+420 555 555 555"
        },
        {
            "element": {
                "id": "6f726c77-36bd-8062-51df-056136e10d35"
            },
            "value": "brnocafe@kontent.ai"
        },
        {
            "element": {
                "id": "5769c0f4-66a8-4c73-3c19-c023bdfa123a"
            },
            "value": [
                {
                    "id": "8fad2e2c-1351-4bfc-be12-007582d61c48"
                }
            ]
        },
        {
            "element": {
                "id": "e82d0f49-5b15-45e1-9b1f-32ccc1be4941"
            },
            "value": [
                {
                    "id": "2ab0aad5-7609-4371-8d6e-cb4a917b2ad1"
                },
                {
                    "id": "405c6578-8233-4277-9826-6b5e74dc6f39"
                }
            ]
        }
    ],
    "workflow_step": {
        "id": "88ac5e6e-1c5c-4638-96e1-0d61221ad5bf"
```

```
    },
    "item": {
        "id": "33389c83-dcfe-48f9-b0ee-f94aeabd2b08"
    },
    "language": {
        "id": "00000000-0000-0000-0000-000000000000"
    },
    "last_modified": "2021-12-03T09:52:05.5604855Z"
}
```

To verify that the variant has been imported, you have two options:

- Retrieve variants of the item using the Management API.
- Check your content list in Kontent.ai.

Avoid security incidents

After you finish importing content, consider deactivating the Management API if you don't plan on using the API further.

What's next?

In this tutorial, you've imported a set of plain text content elements into a new content item. You can use the same process to import any number of more complex content items.

- Learn to import rich text, linked content (when importing content items that reference each other), or assets.
- Check out descriptions of all elements in content types and all elements within language variants in the API reference when importing other content elements.
- Validate the content of your project to check for possible issues, for example, references to missing content items after your import process is finished.
- Check out other tools for content migration.