Get content items

November 23, 2021 • Jan Cerman • 4 min read • JavaScript

While your copywriters draft articles and add finishing touches to the content in your project, deliver that content to web and mobile applications using the Delivery API. The Delivery API is a read-only API that's available as both REST API and GraphQL API. This tutorial covers the basics of using the Delivery REST API.

The Delivery API can work in two modes: production and preview. In production mode, you get published content that is publicly available. In preview mode, you get the latest version of your content, be it published or unpublished.

Let's find out what you need to retrieve a list of published content items like articles from your project.

Get content items

To retrieve content items from a project, you first need to specify the project using its unique ID.

You'll use the project ID to tell the Delivery REST API where to look for content. For example, a project ID might look like this: `8d20758c-d74c-4f59-ae04-ee928c0816b`.

1. Find your project ID

1. In Kontent, choose a project.
2. From the app menu, choose **Project settings.**
3. Under **Environment settings**, choose **API keys**.
4. In the **Delivery API** box, click .

With the project ID, you can now make queries to the Delivery API.

2. Make a request

Calling the list content items endpoint gives you all content items from the specified project in the JSON format.

JavaScript

```javascript
// Tip: Find more about JS/TS SDKs at https://docs.kontent.ai/javascript
const KontentDelivery = require('@kentico/kontent-delivery');

const deliveryClient = KontentDelivery.createDeliveryClient({
  projectId: '8d20758c-d74c-4f59-ae04-ee928c0816b7'
});

const response = await deliveryClient.items()
  .toPromise();
```

Note that recently published items may appear in the Delivery API after a slight delay.

Paging the results

If you don't need all content items at once, you can tinker with the paging by specifying the `limit` and `skip` query parameters.

For example, calling the `/items` endpoint with the `limit=3&skip=6` query parameters sets the page size to 3 and gives you the third page.

Filter content items

Now that you can get all content items from your project, you need to filter them to get only a specific few. In this example, you'll retrieve articles. These are the content items based on the *Article* content type.

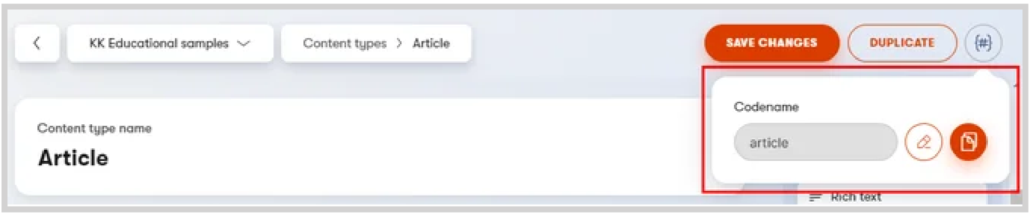
1. Find codenames

To move any further, you need to find the **codename** of the *Article* content type.

Quick facts about codenames

Codenames are identifiers of objects in Kontent. A codename is initially generated by the system from the object's name when it's saved for the first time.

You can copy codenames by clicking {#} near the name of a content type, content element, or other objects in your project. For example, the find the codename of a content type named *Article*, go to Content model > Content types > Article > {#}.



Example: Displaying the codename of the Article content type.

Once you have the codename (in this case `article`) you can use it to filter the requested content items by their type.

2. Filter by codenames

The information about a content item's type is stored in the content item's *System* object, specifically, in its `type` property. The *System* object contains metadata about the content item such as the last content modification date, language, collection, and more.

JSON

```json
"system": {
  "id": "31f8470f-8a94-438a-8a47-f4cdb9c90ada",
  "collection": "default",
  "name": "Why structured writing needs structured content",
  "codename": "structured_writing",
  "language": "en-US",
  "type": "article",
  "sitemap_locations": [],
  "last_modified": "2020-01-27T13:43:47.134249Z",
  "workflow_step": "published"
}
```

To filter the content items by type, you need to compare the value in the `type` system property to `article` using the following notation: `system.type=article`. Any content items that are not based on the *Article* content type will be omitted from the response.

JavaScript

```javascript
// Tip: Find more about JS/TS SDKs at https://docs.kontent.ai/javascript
const KontentDelivery = require('@kentico/kontent-delivery');

const deliveryClient = KontentDelivery.createDeliveryClient({
  projectId: '8d20758c-d74c-4f59-ae04-ee928c0816b7'
});

const response = await deliveryClient.items()
  .type('article')
  .toPromise();
```

The value comparison is done using the equals operator.

Tip: Check out more examples of filtering content with the Delivery APIs.

Order content items

The Delivery API sorts content items alphabetically by their codenames by default. But with content like articles, you usually want to retrieve and display them in a certain order and get, for example, only three latest articles from your project.

When getting lists of content items, you can specify their order by using the `order` query parameter. The value of the `order` query parameter must be in the following format: `<PropertyToOrderBy>[<asc|desc>]`. Where the `PropertyToOrderBy` value specifies either a System property (such as `system.type`) or a content element within a content item (such as `elements.title`). For instance, if you don't specify the order when retrieving content, it is the equivalent of adding `order=system.codename[asc]` to your query.

To get three latest articles from your project, you need to provide the following query parameters:

- `system.type=article` – specifies the content type of the content items.
- `limit=3` – sets the number of content items to return (sometimes also referred to as page size).
- `order=system.last_modified[desc]` – sorts the content items by last modification date in descending order.

JavaScript

```javascript
// Tip: Find more about JS/TS SDKs at https://docs.kontent.ai/javascript
const KontentDelivery = require('@kentico/kontent-delivery');

const deliveryClient = KontentDelivery.createDeliveryClient({
  projectId: '8d20758c-d74c-4f59-ae04-ee928c0816b7'
});

const response = await deliveryClient.items()
  .type('article')
  .limitParameter(3)
  .orderParameter('system.last_modified', KontentDelivery.SortOrder.desc)
  .toPromise();
```

What's next?

You've learned how to get specific content from your Kontent project with filtering and sorting. Besides fetching content items, you can also use the Delivery REST API to get content types, elements, and taxonomies.

- Future-proof your app with best practices on getting content.
- Set up content preview so that editors can preview unpublished content.
- Map your project's content types to strongly typed models to streamline your development process.
- Share content between projects either directly in the UI or programmatically with Delivery SDKs.

Want to create an SDK for your preferred technology? Check out our guidelines for SDK developers.