Exercise:
Build a simple blog managed by Contentful

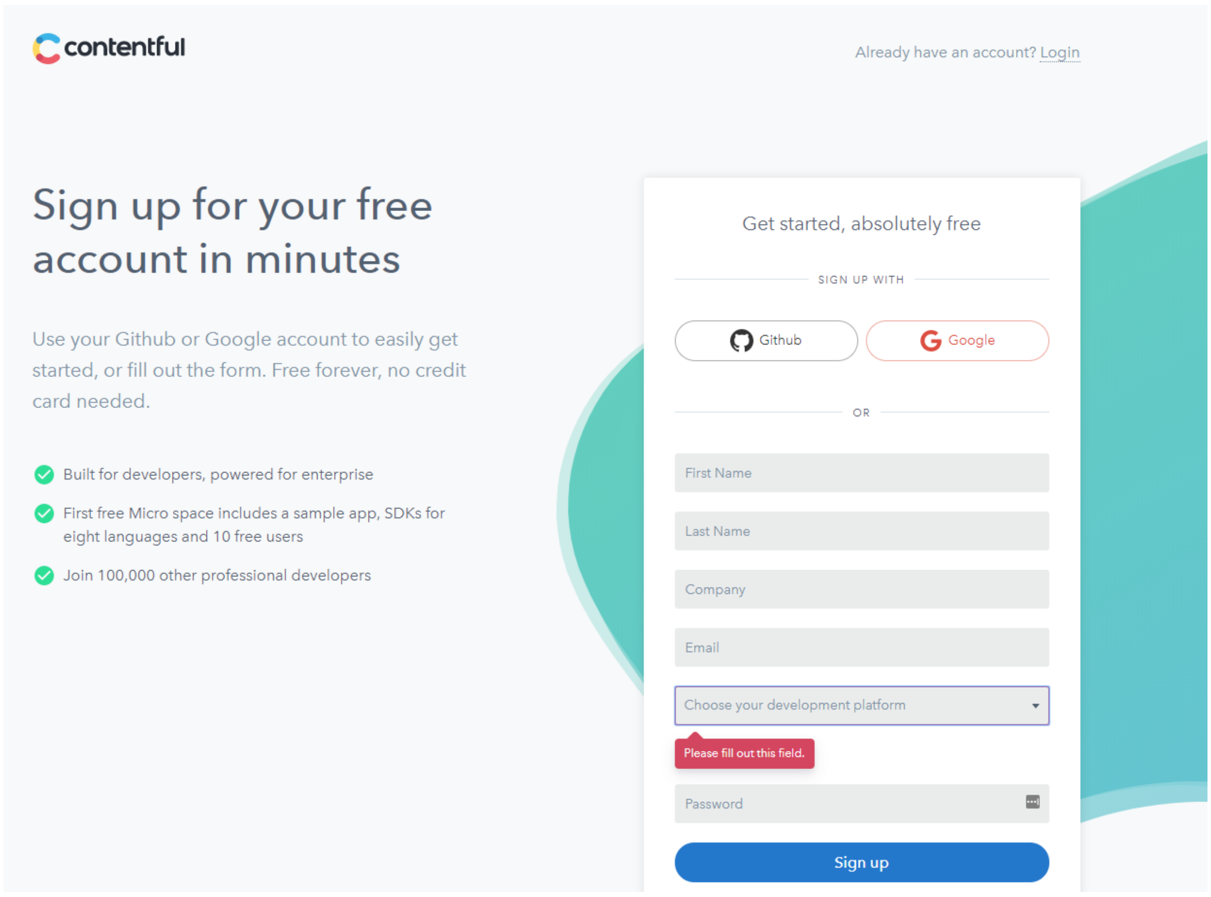
The goal of this exercise is to learn about basic content management by configuring a simple content model for a blog post, creating a blog post based on that content model and finally connect it to a template made in HTML/CSS/JS.

First step: Sign up with Contentful

There are many CMS vendors out there. You might have heard of Wordpress, Umbraco, Drupal, Sitecore, Joomla or even Episerver. But to keep this exercise as simple and ‘developer friendly’ as possible, we’ll use Contentful, a completely cloud based SaaS system, where you can easily create a free developer account.

Contentful works as a “ContentHub” or “Headless CMS” meaning that it’s purely storing content and providing an editorial interface for it. Developers can then take the content and use it through a powerful API and connectors to most major programming platforms. That way you can use the content in both apps, static generated sites or regular backend driven web sites.

First, go to https://www.contentful.com/sign-up/



Sign up for your free account in minutes

Use your Github or Google account to easily get started, or fill out the form. Free forever, no credit card needed.

- Built for developers, powered for enterprise
- First free Micro space includes a sample app, SDKs for eight languages and 10 free users
- Join 100,000 other professional developers

Get started, absolutely free

SIGN UP WITH

Github | Google

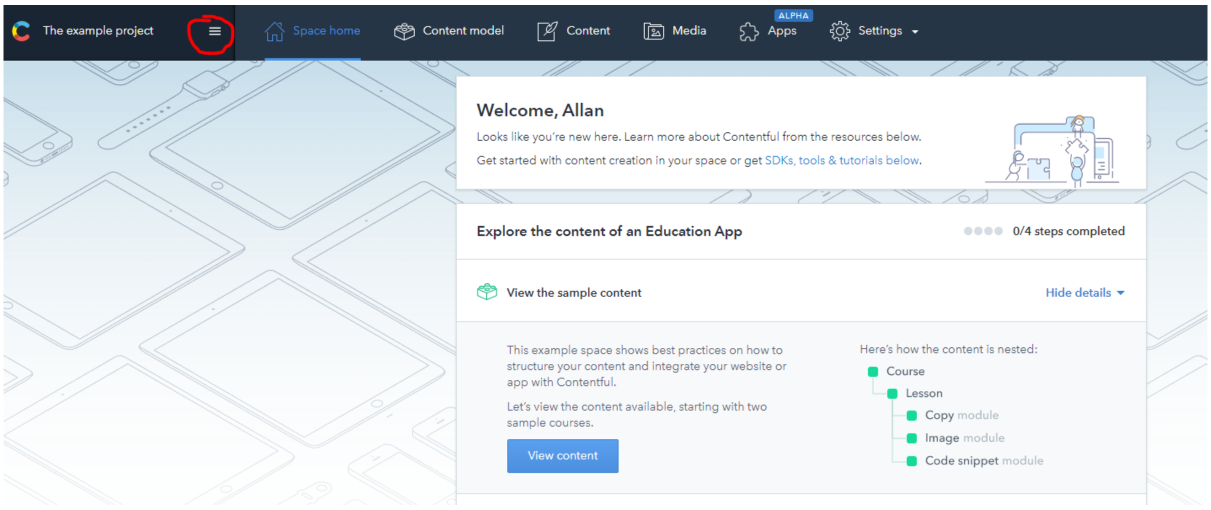
OR

First Name
Last Name
Company
Email
Choose your development platform
Please fill out this field.
Password

Sign up

Here you can sign up – either fast and easy using Github or Google credentials, or with your email and password.

Setting up an empty space



Welcome, Allan

Looks like you're new here. Learn more about Contentful from the resources below.
Get started with content creation in your space or get SDKs, tools & tutorials below.

Explore the content of an Education App — 0/4 steps completed

View the sample content — Hide details

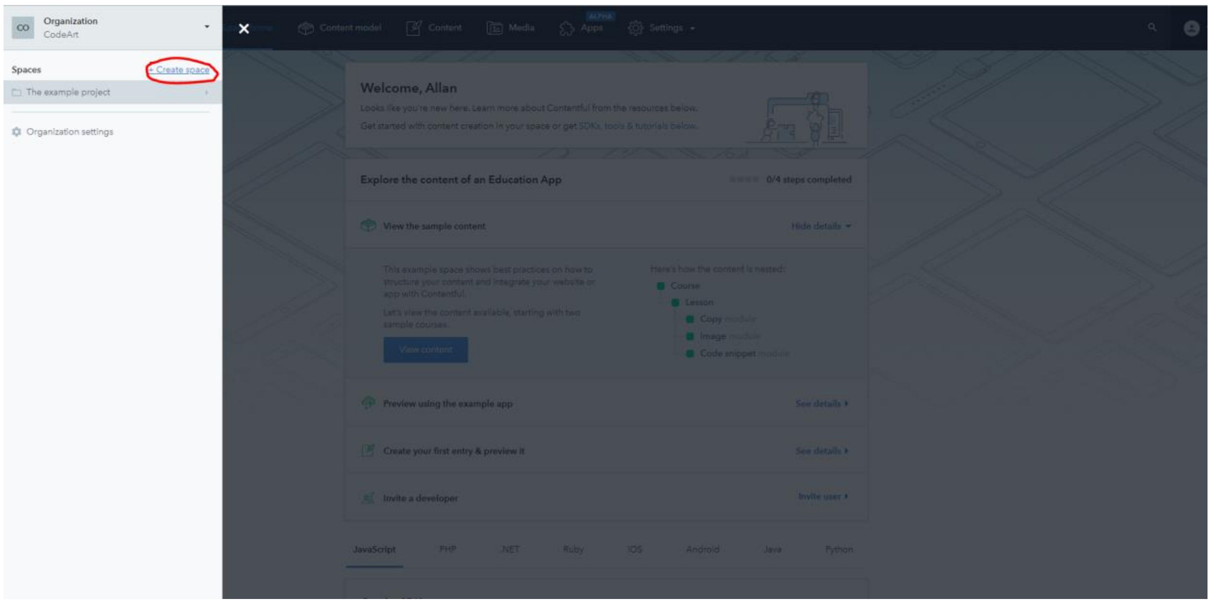
This example space shows best practices on how to structure your content and integrate your website or app with Contentful.

Let's view the content available, starting with two sample courses.

View content

Here's how the content is nested:
- Course
  - Lesson
    - Copy module
    - Image module
    - Code snippet module

Once you have signed up, it will have started by setting up a sample project (called 'Space') for you. However, we don't need that right now, so instead click the menu next to "Space Home" and create a new space.



Spaces + Create space

The example project

Organization settings

Welcome, Allan

Looks like you're new here. Learn more about Contentful from the resources below.

Get started with content creation in your space or get SDKs, tools & tutorials below.

Explore the content of an Education App 0/4 steps completed

View the sample content Hide details

This example space shows best practices on how to structure your content and integrate your website or app with Contentful.

Let's view the content available, starting with two sample courses.

View content

Here's how the content is nested:

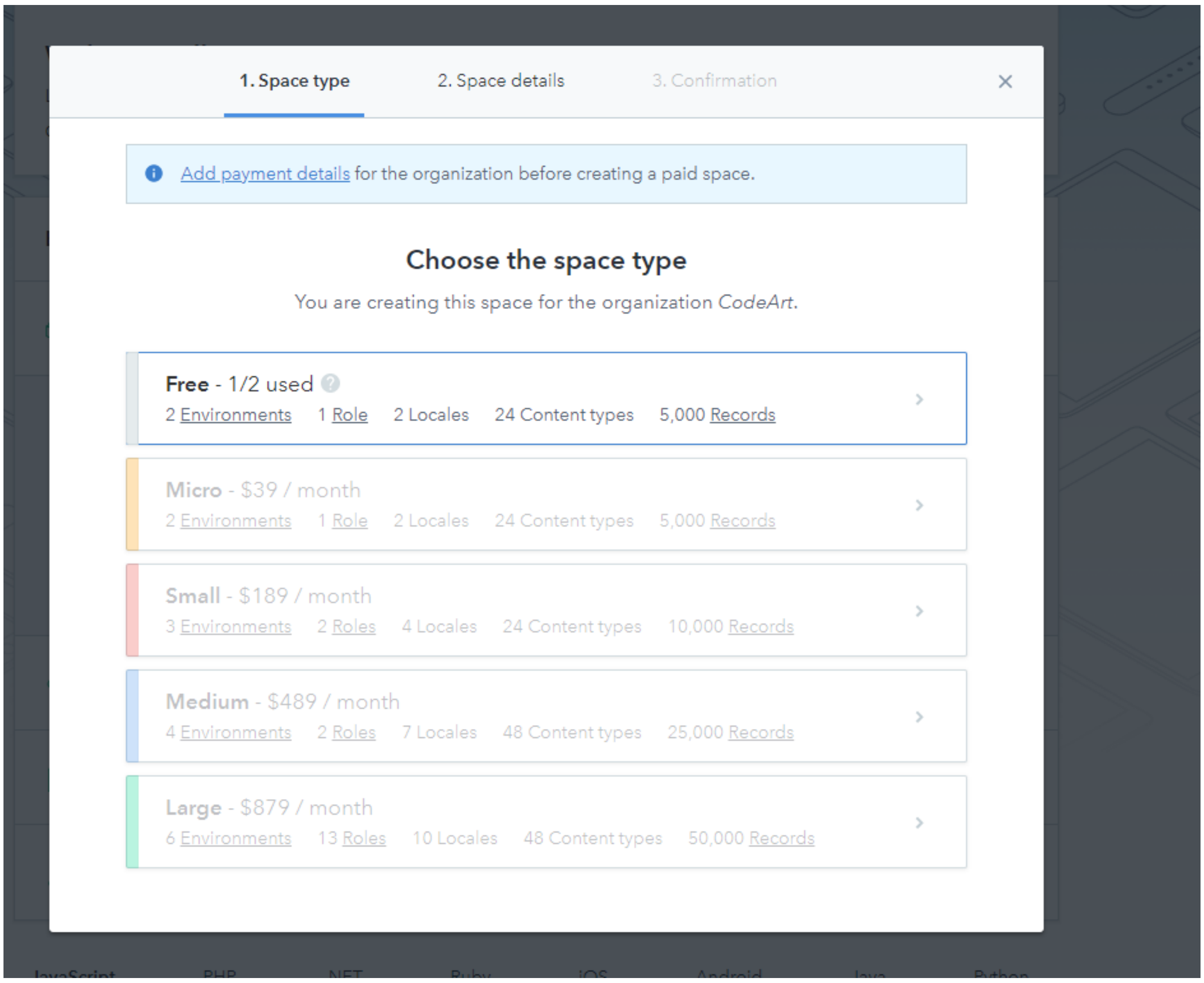
- Course
  - Lesson
    - Copy module
    - Image module
    - Code snippet module

Preview using the example app See details

Create your first entry & preview it See details

Invite a developer Invite user

JavaScript PHP .NET Ruby iOS Android Java Python



Add payment details for the organization before creating a paid space.

Choose the space type

You are creating this space for the organization *CodeArt*.

Free - 1/2 used
2 Environments 1 Role 2 Locales 24 Content types 5,000 Records

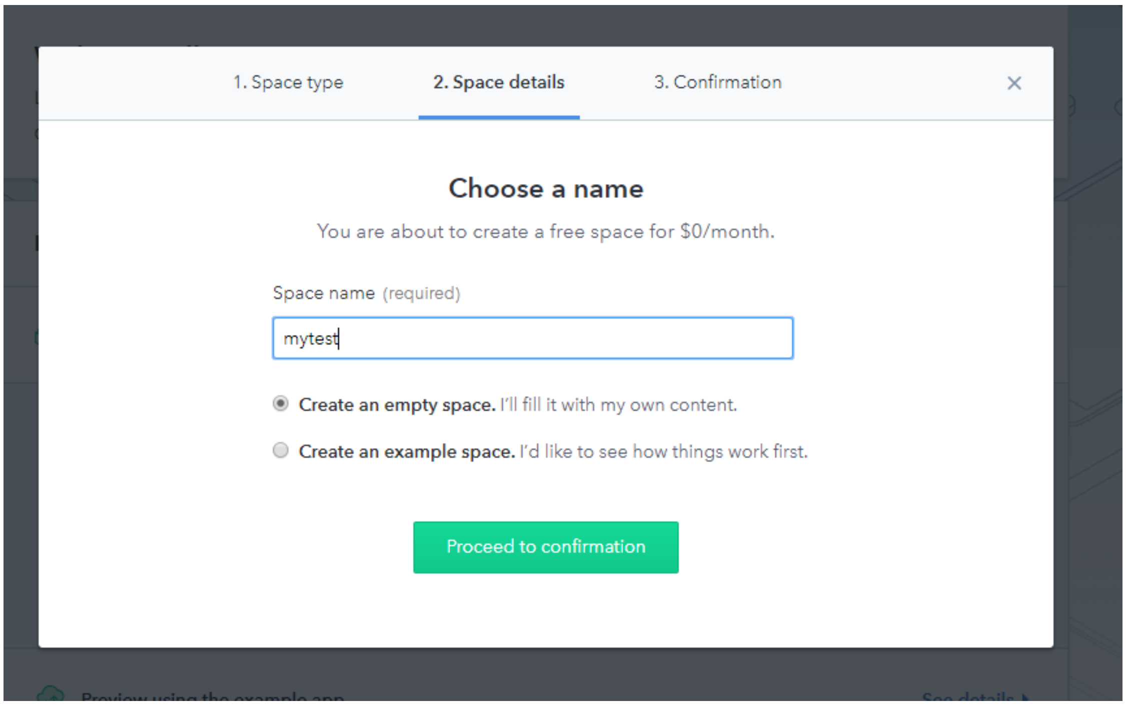
Micro - $39 / month
2 Environments 1 Role 2 Locales 24 Content types 5,000 Records

Small - $189 / month
3 Environments 2 Roles 4 Locales 24 Content types 10,000 Records

Medium - $489 / month
4 Environments 2 Roles 7 Locales 48 Content types 25,000 Records

Large - $879 / month
6 Environments 13 Roles 10 Locales 48 Content types 50,000 Records

Obviously, we'll pick a free space for now (don't worry, you can have 2 of them). No need to add payment information at this time.



Choose a name

You are about to create a free space for $0/month.

Space name (required): mytest

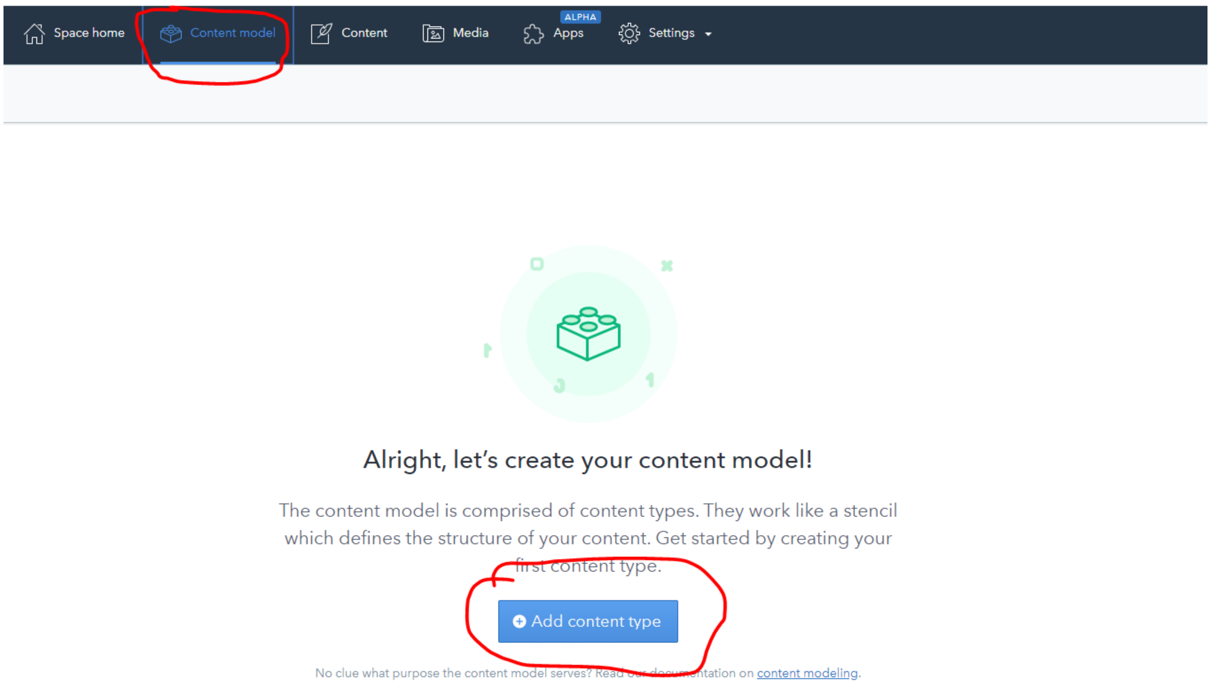
[✓ Create an empty space.] I'll fill it with my own content.

Create an example space. I'd like to see how things work first.

Give the space a name and choose “Empty space”. Go ahead and confirm it and it will be created.

Defining the content model

Now, it’s time to define the content model(s) we need – in this case we’ll just start simple with 1 content type, the “Blog Post”.

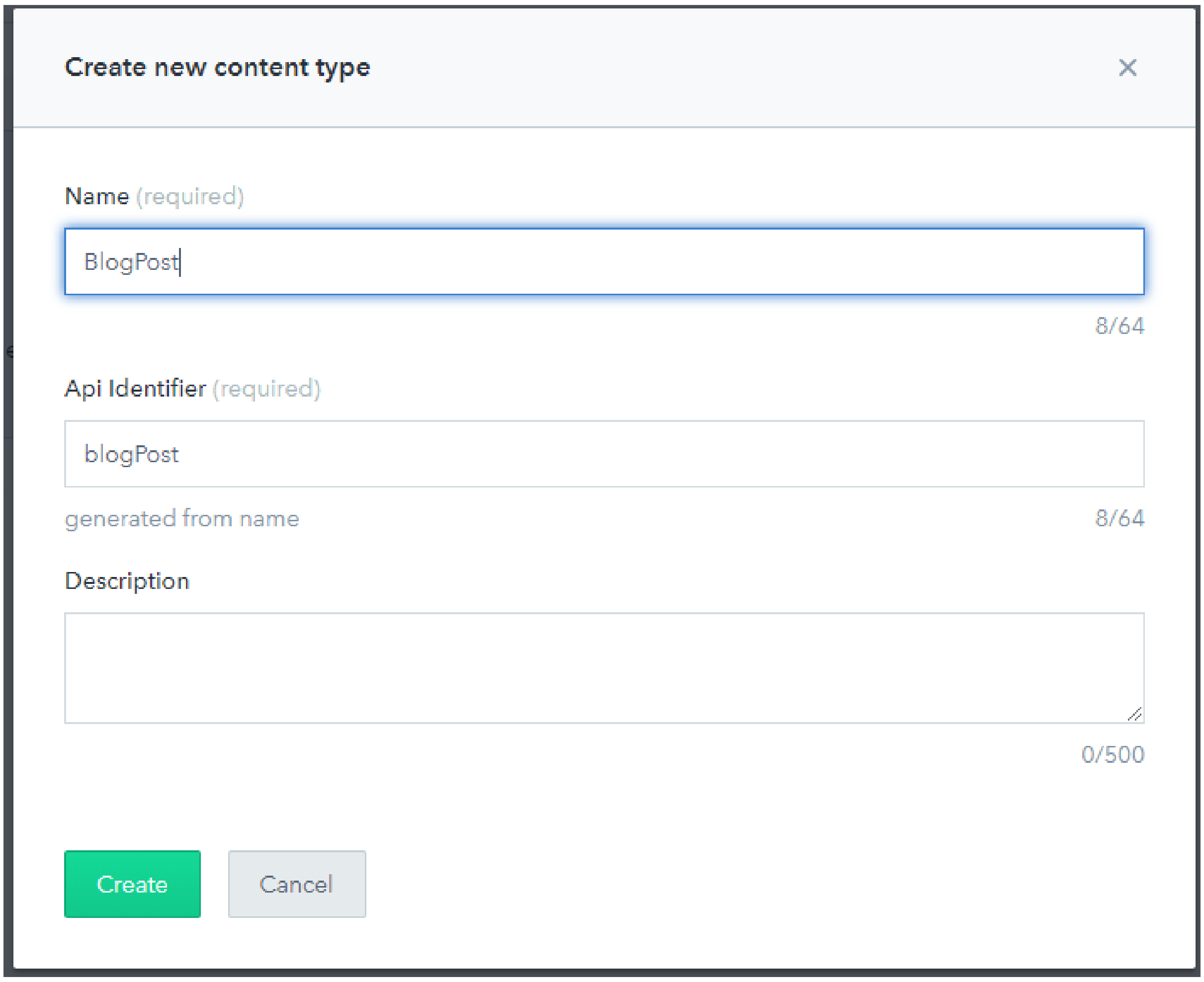


Alright, let's create your content model!

The content model is comprised of content types. They work like a stencil which defines the structure of your content. Get started by creating your first content type.

No clue what purpose the content model serves? Read our documentation on content modeling.

Go to the “Content Model” in the top navigation and then click “Add content type”.



Create new content type

Name (required)

BlogPost

8/64

Api Identifier (required)

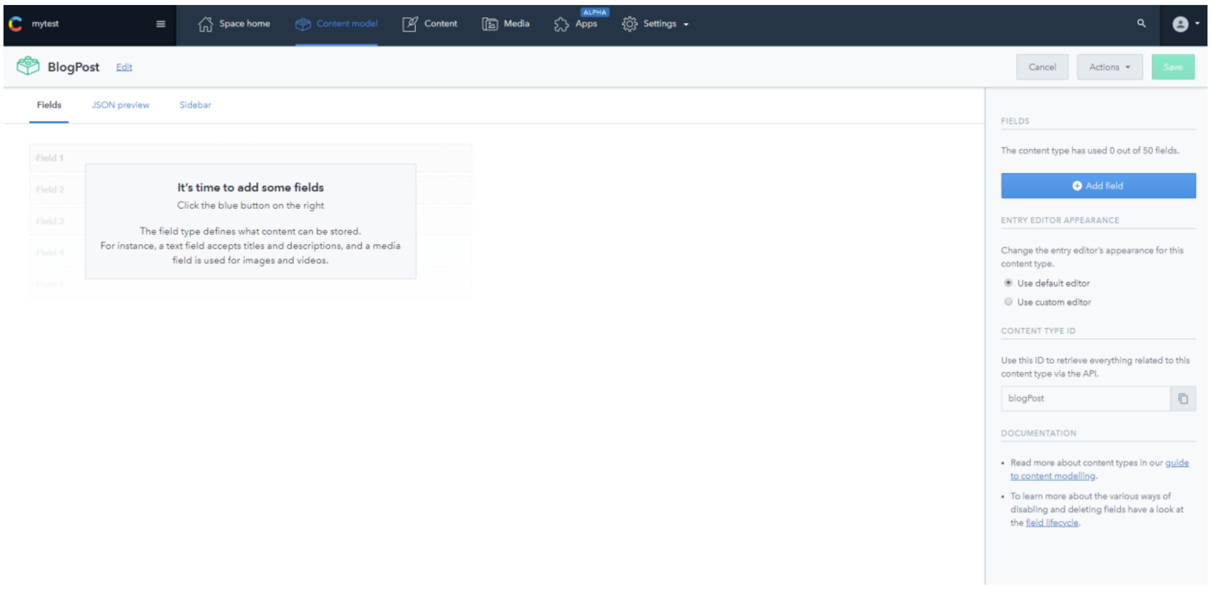
blogPost

generated from name 8/64

Description

0/500

Once created, we should give it a name (and if you want a description).



It's time to add some fields

Click the blue button on the right

The field type defines what content can be stored.
For instance, a text field accepts titles and descriptions, and a media field is used for images and videos.

FIELDS

The content type has used 0 out of 50 fields.

Add field

ENTRY EDITOR APPEARANCE

Change the entry editor's appearance for this content type.

- ● Use default editor
- ○ Use custom editor

CONTENT TYPE ID

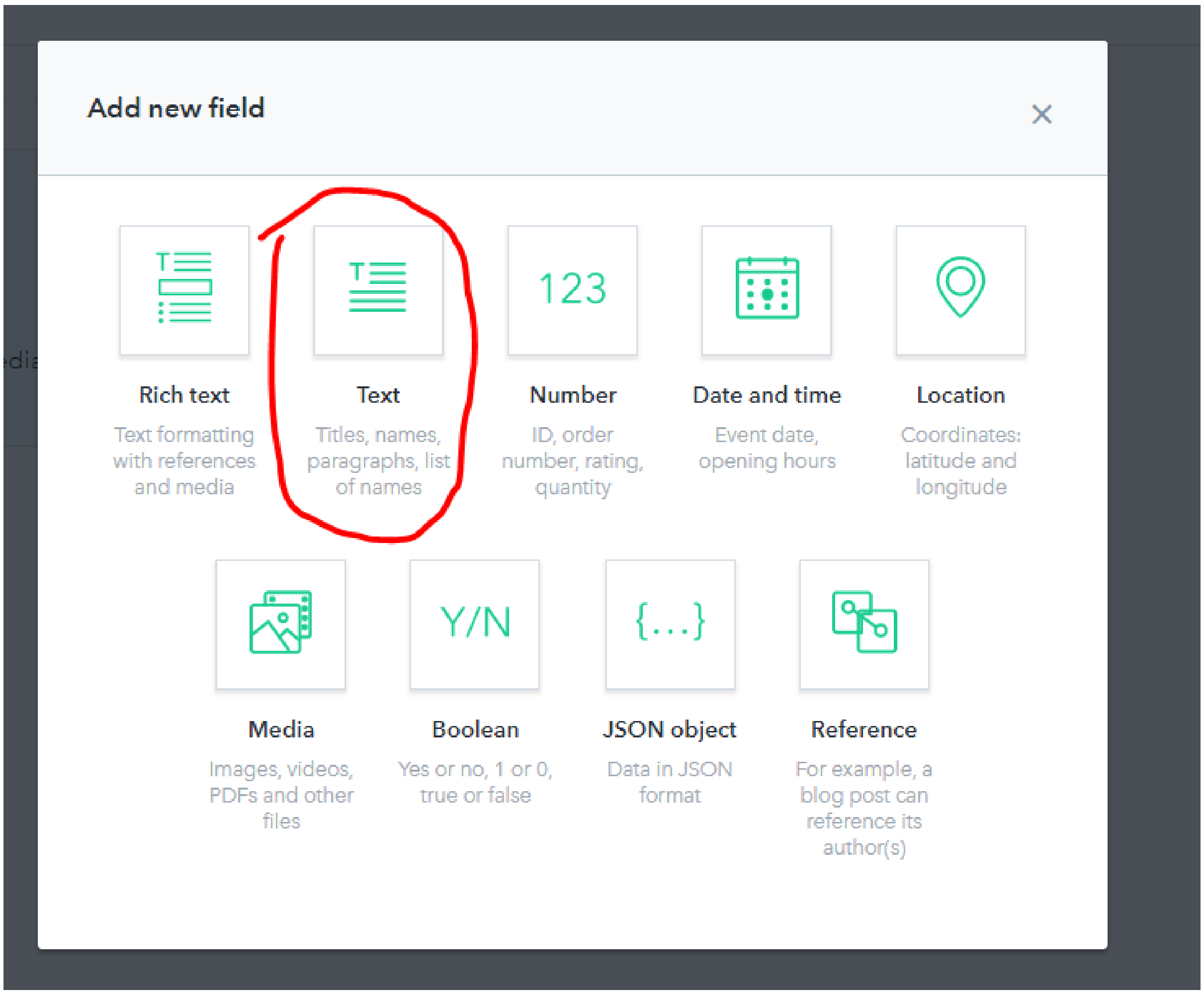
Use this ID to retrieve everything related to this content type via the API.

blogPost

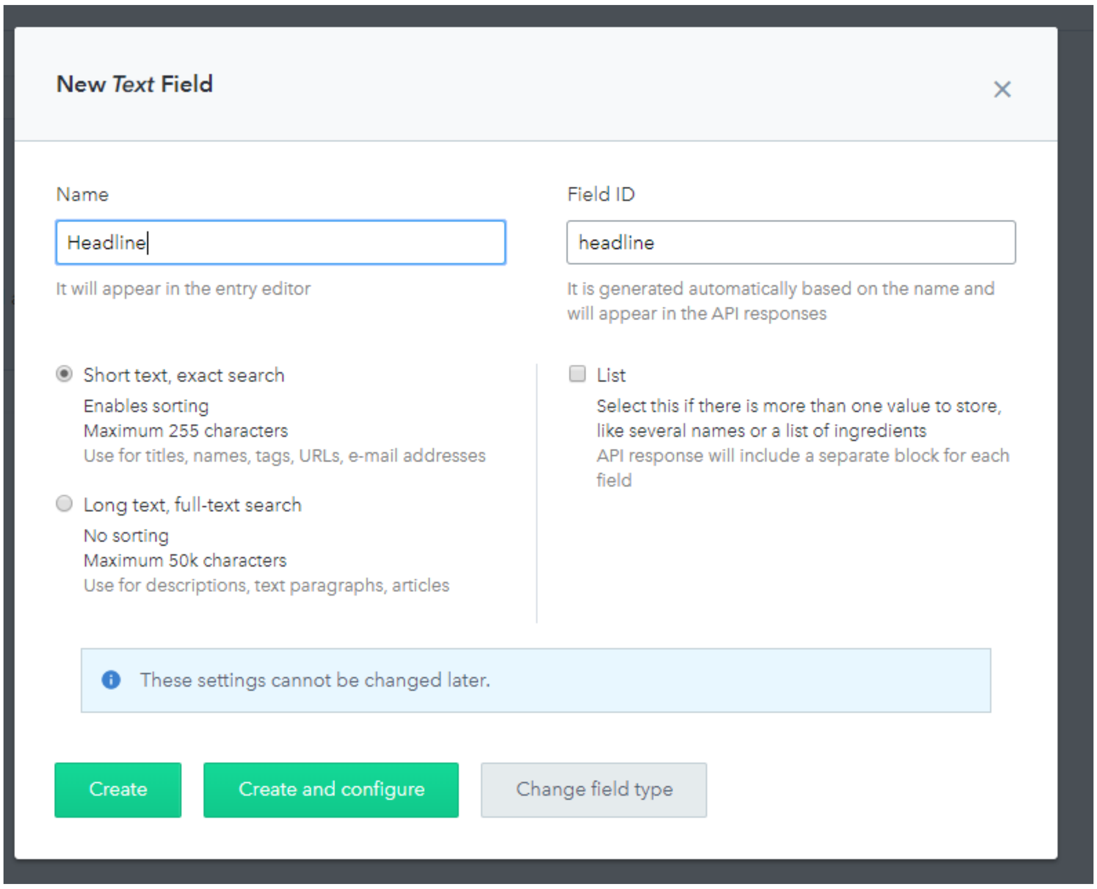
DOCUMENTATION

- Read more about content types in our guide to content modelling.
- To learn more about the various ways of disabling and deleting fields have a look at the field lifecycle.

Now, it's time to start adding fields/properties to the content type. This is done by clicking the "Add field" on the right hand side.


Add new field
Rich text
Text formatting with references and media
Text
Titles, names, paragraphs, list of names
123
Number
ID, order number, rating, quantity
Date and time
Event date, opening hours
Location
Coordinates: latitude and longitude
Media
Images, videos, PDFs and other files
Y/N
Boolean
Yes or no, 1 or 0, true or false
{...}
JSON object
Data in JSON format
Reference
For example, a blog post can reference its author(s)

For starters we'll create a "Text" field to hold the title/headline of the blog post.


New *Text* Field

Name: Headline
It will appear in the entry editor

Field ID: headline
It is generated automatically based on the name and will appear in the API responses

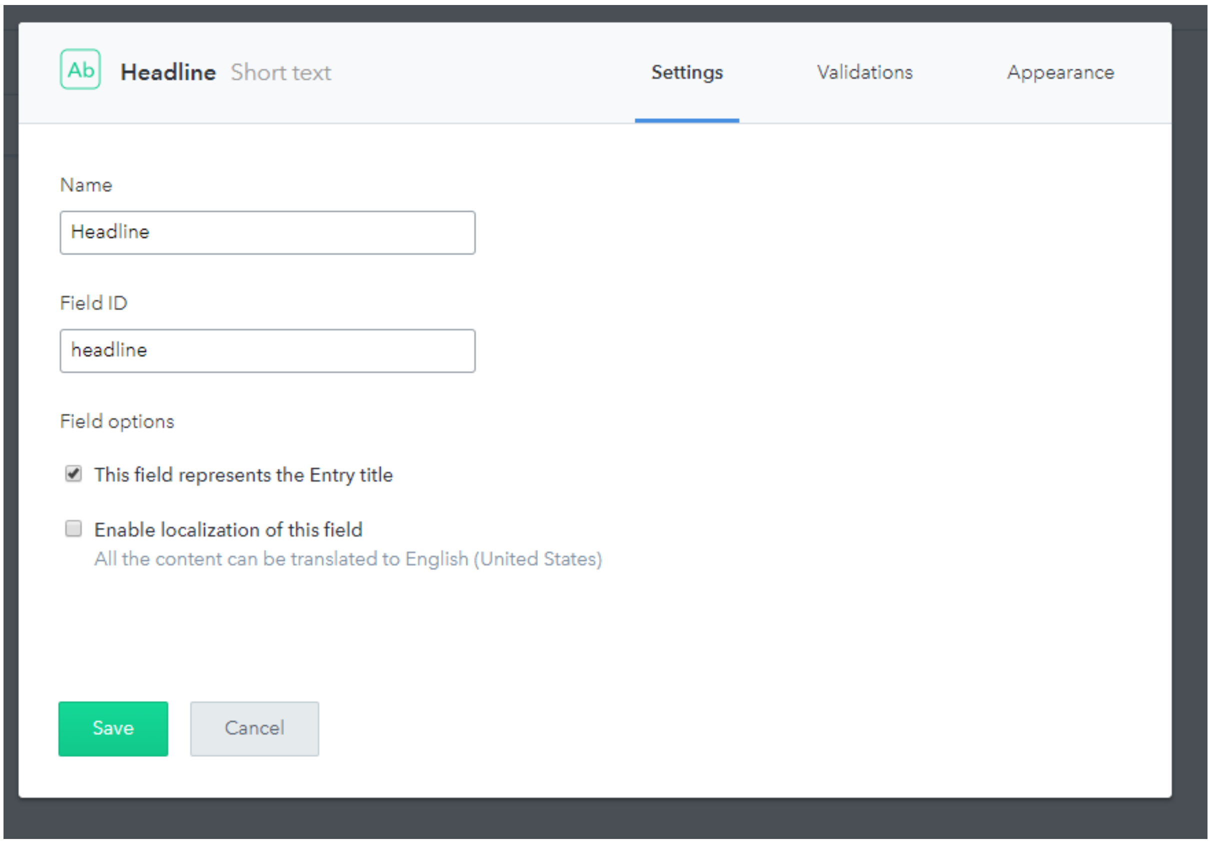
[✓ Short text, exact search]
Enables sorting
Maximum 255 characters
Use for titles, names, tags, URLs, e-mail addresses

Long text, full-text search
No sorting
Maximum 50k characters
Use for descriptions, text paragraphs, articles

☐ List
Select this if there is more than one value to store, like several names or a list of ingredients
API response will include a separate block for each field

These settings cannot be changed later.

We give it a name. In this case “Headline”. Note that there is also automatically a field ID created in lowercase. We’ll need that later. Click “Create and Configure”.



Name: Headline

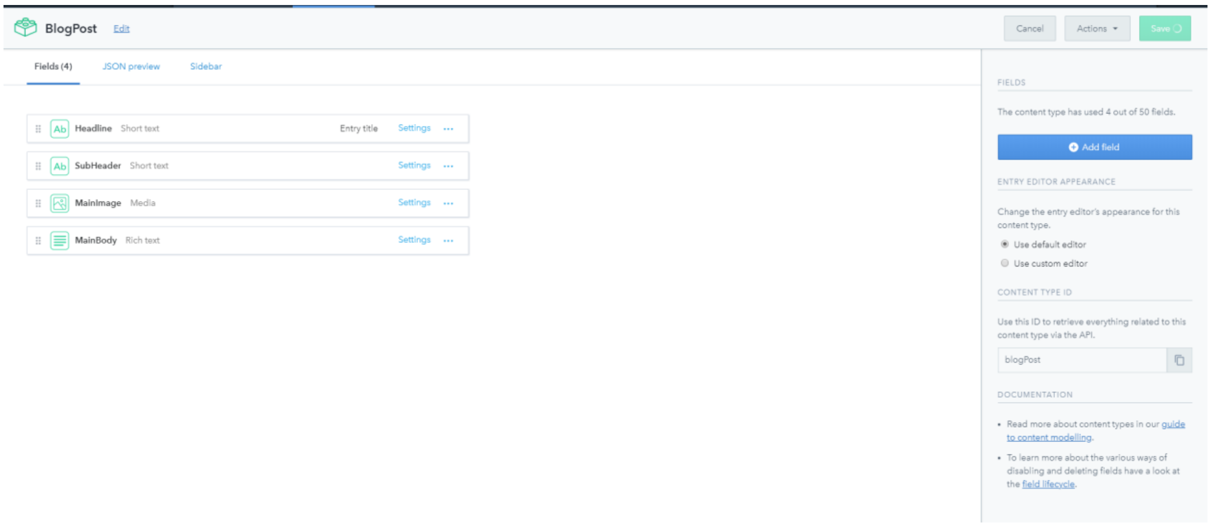
Field ID: headline

Field options

☑ This field represents the Entry title

☐ Enable localization of this field
All the content can be translated to English (United States)

At this point we can set up various custom settings for the headline field. Let's just check the box to indicate that it's the Title and save it.



BlogPost Edit

Fields (4) | JSON preview | Sidebar

Headline	Short text	Entry title	Settings
SubHeader	Short text		Settings
MainImage	Media		Settings
MainBody	Rich text		Settings

FIELDS

The content type has used 4 out of 50 fields.

ENTRY EDITOR APPEARANCE

Change the entry editor's appearance for this content type.

[✓ Use default editor] Use custom editor

CONTENT TYPE ID

Use this ID to retrieve everything related to this content type via the API.

blogPost

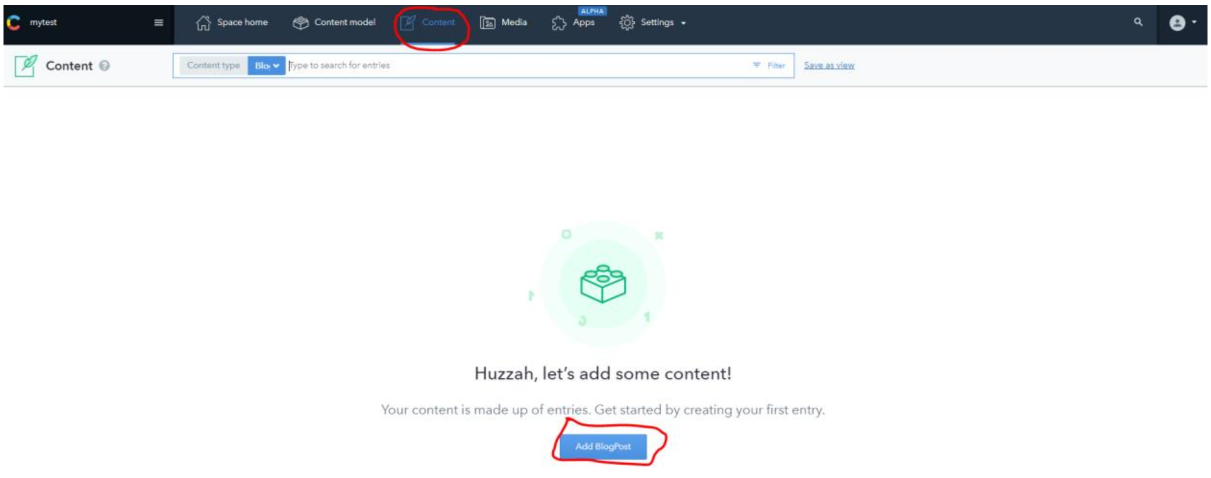
DOCUMENTATION

- Read more about content types in our guide to content modelling.
- To learn more about the various ways of disabling and deleting fields have a look at the field lifecycle.

After that is done, we can add multiple other fields. I suggest a SubHeader, an image (called "MainImage") and a MainBody. Note that they have different types. The image for example is a Media – meaning it holds a reference to a media item, and the MainBody is rich text edited using a rich text editor.

Add them one at a time and when you are done with it all, save the content model. You can of course always go back and add more.

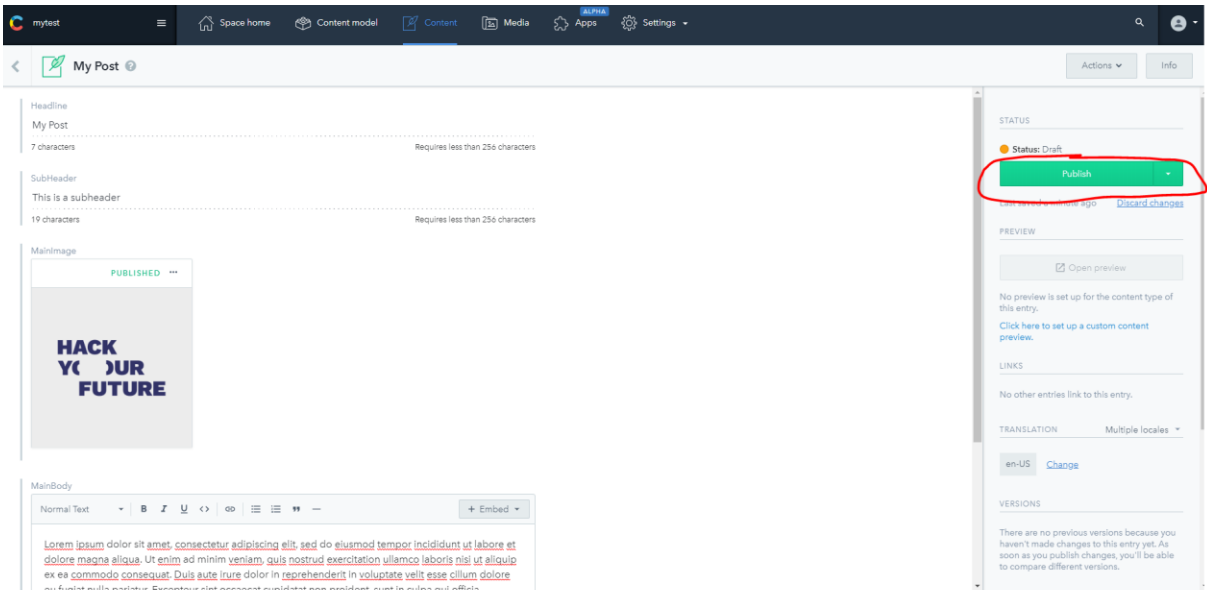
Creating Content



Huzzah, let's add some content!

Your content is made up of entries. Get started by creating your first entry.

It's now time to start adding some content, so let's click "Content" in the top menu, followed by "Add BlogPost".



My Post

Headline
My Post
7 characters Requires less than 256 characters

SubHeader
This is a subheader
19 characters Requires less than 256 characters

MainImage

MainBody
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore

STATUS
Status: Draft
Last saved a minute ago Discard changes

PREVIEW
No preview is set up for the content type of this entry.
Click here to set up a custom content preview.

LINKS
No other entries link to this entry.

TRANSLATION Multiple locales
en-US

VERSIONS
There are no previous versions because you haven't made changes to this entry yet. As soon as you publish changes, you'll be able to compare different versions.

Now we are working as an editor would typically work, creating and managing content items. We can see all the fields we defined earlier in the content model and we can start filling in some content.

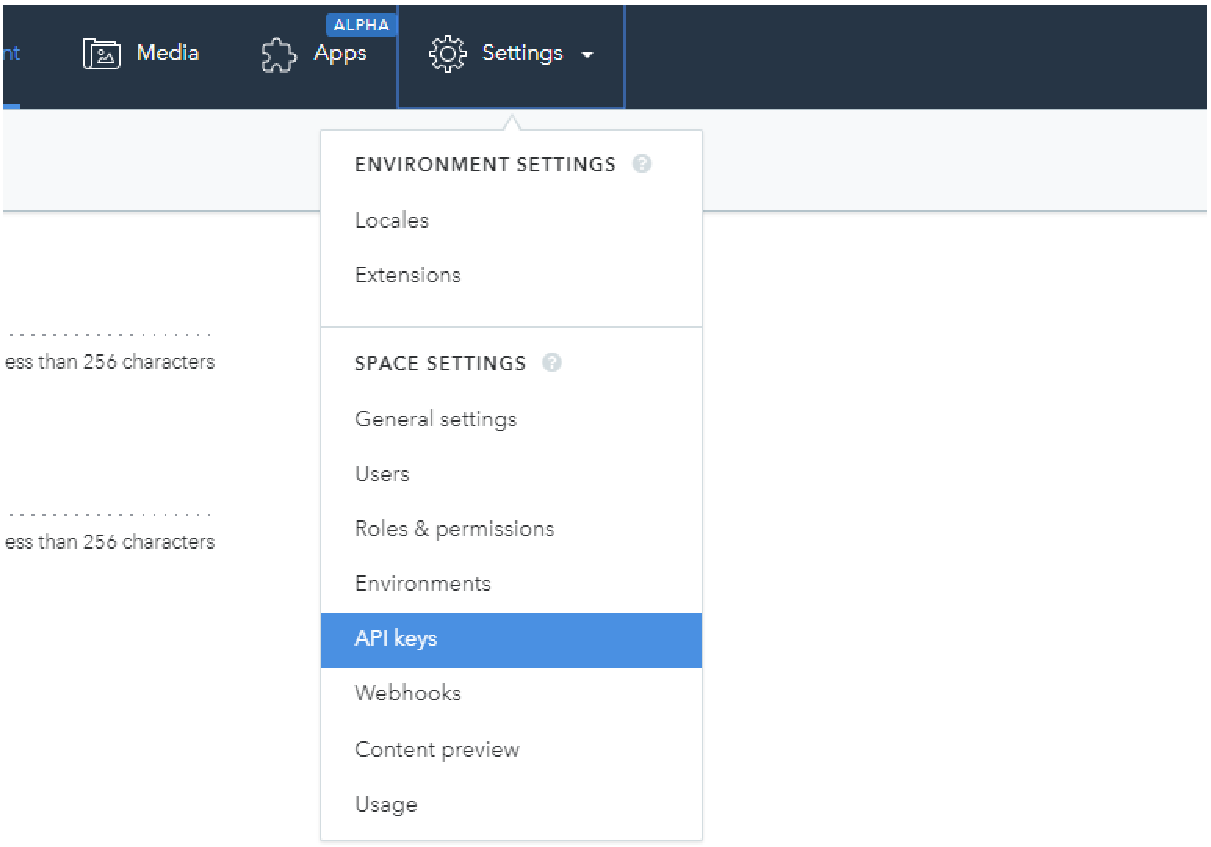
Add some text to the various fields – and for the image, you can “Add and link” a media item, where you upload an image you have locally. Once you are ready click “Publish”.

Feel free to add multiple blog posts if you want.

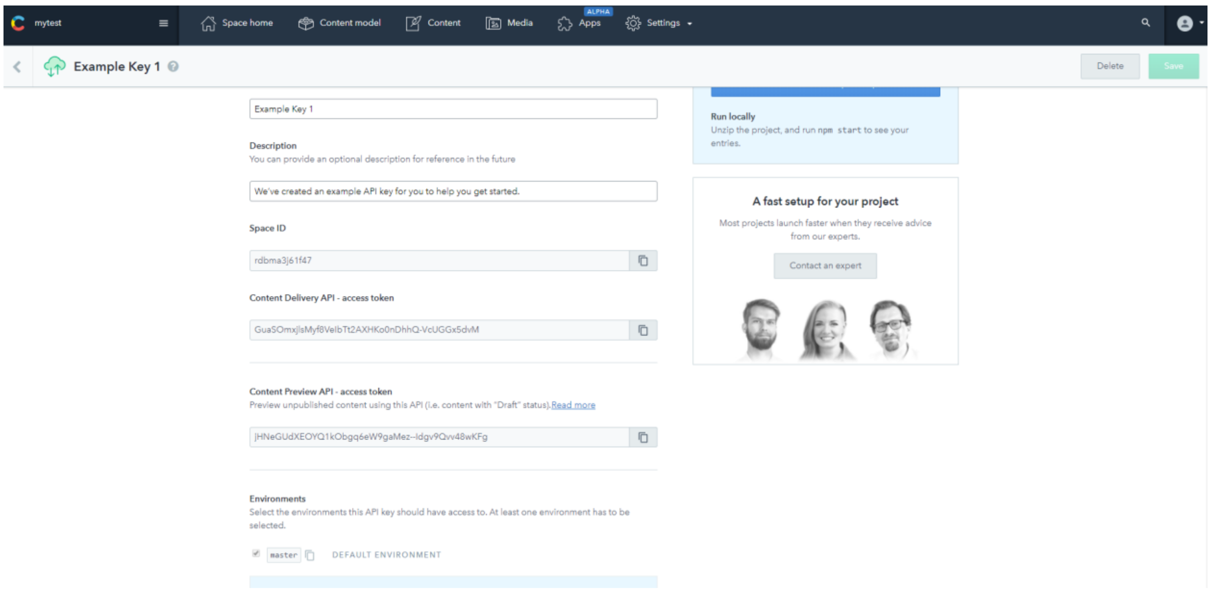
Rendering the blog post to a web site visitor

Now that we have created a content model and some sample content it’s time we try to produce an html template that can be used to render this content dynamically.

Since Contentful has a wide range of connectors, we could do this natively in NodeJS, React, Php, .NET, Python or any other number of languages and platforms. To keep it simple and understandable and within HYF curriculum the following example is based on plain Javascript, but feel free to use whatever platform you prefer. There is good documentation for each connector in Contentful.



First off, we need to get the API keys in order to connect with the API. So we go to “Settings” and API keys”. There should already be a set of keys called “Example Key” that we can use.



Example Key 1

Description
You can provide an optional description for reference in the future

We've created an example API key for you to help you get started.

Space ID

rdbma3j61f47

Content Delivery API - access token

GuaSOmxjlsMyf8VelbTt2AXHKo0nDhhQ-VcUGGx5dvM

Content Preview API - access token
Preview unpublished content using this API (i.e. content with "Draft" status). Read more

jHNeGUdXEOYQ1kObgq6eW9gaMez--Idgv9Qvv48wKFg

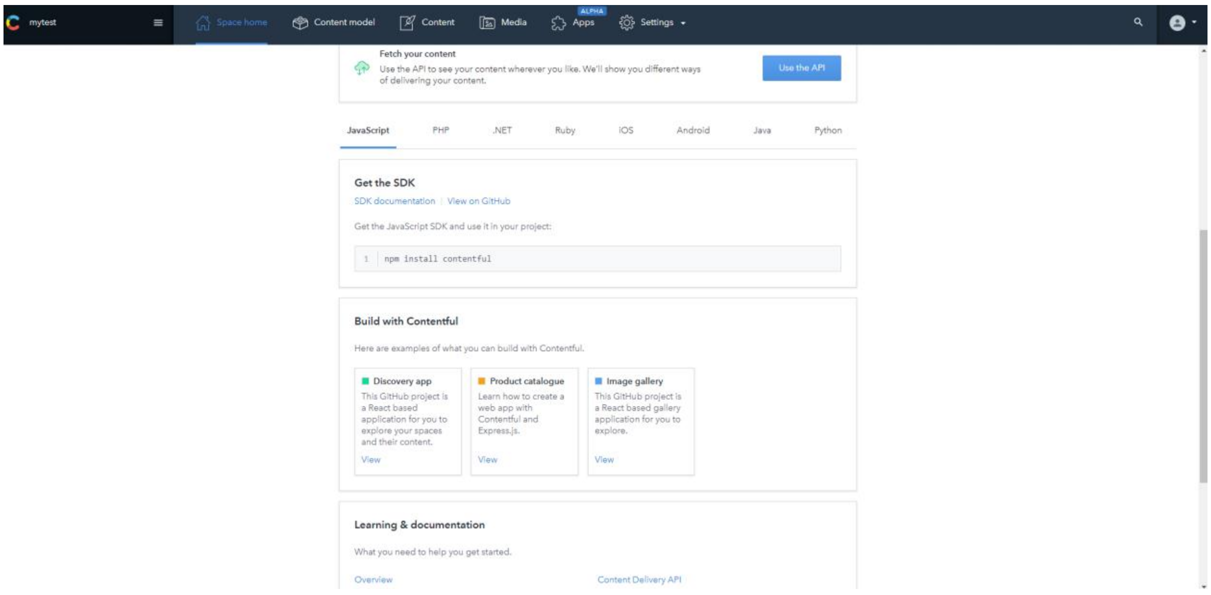
Environments
Select the environments this API key should have access to. At least one environment has to be selected.

✓ master DEFAULT ENVIRONMENT

Run locally
Unzip the project, and run `npm start` to see your entries.

A fast setup for your project
Most projects launch faster when they receive advice from our experts.

We’ll need the SpaceID and the “Content Delivery API access token” for now.



Fetch your content

Use the API to see your content wherever you like. We'll show you different ways of delivering your content.

Get the SDK

Get the JavaScript SDK and use it in your project:

```
npm install contentful
```

Build with Contentful

Here are examples of what you can build with Contentful.

Discovery app
This GitHub project is a React based application for you to explore your spaces and their content.

Product catalogue
Learn how to create a web app with Contentful and Express.js.

Image gallery
This GitHub project is a React based gallery application for you to explore.

Learning & documentation

What you need to help you get started.

If you go to "Space Home" you'll see a list of various platforms and instructions on how to get started with them.

You will find some good introductions to using javascript here:

https://github.com/contentful/contentful.js

https://www.contentful.com/developers/docs/javascript/

The following steps are loosely based on this in-depth guide:
https://www.contentful.com/developers/docs/javascript/tutorials/using-js-cda-sdk/

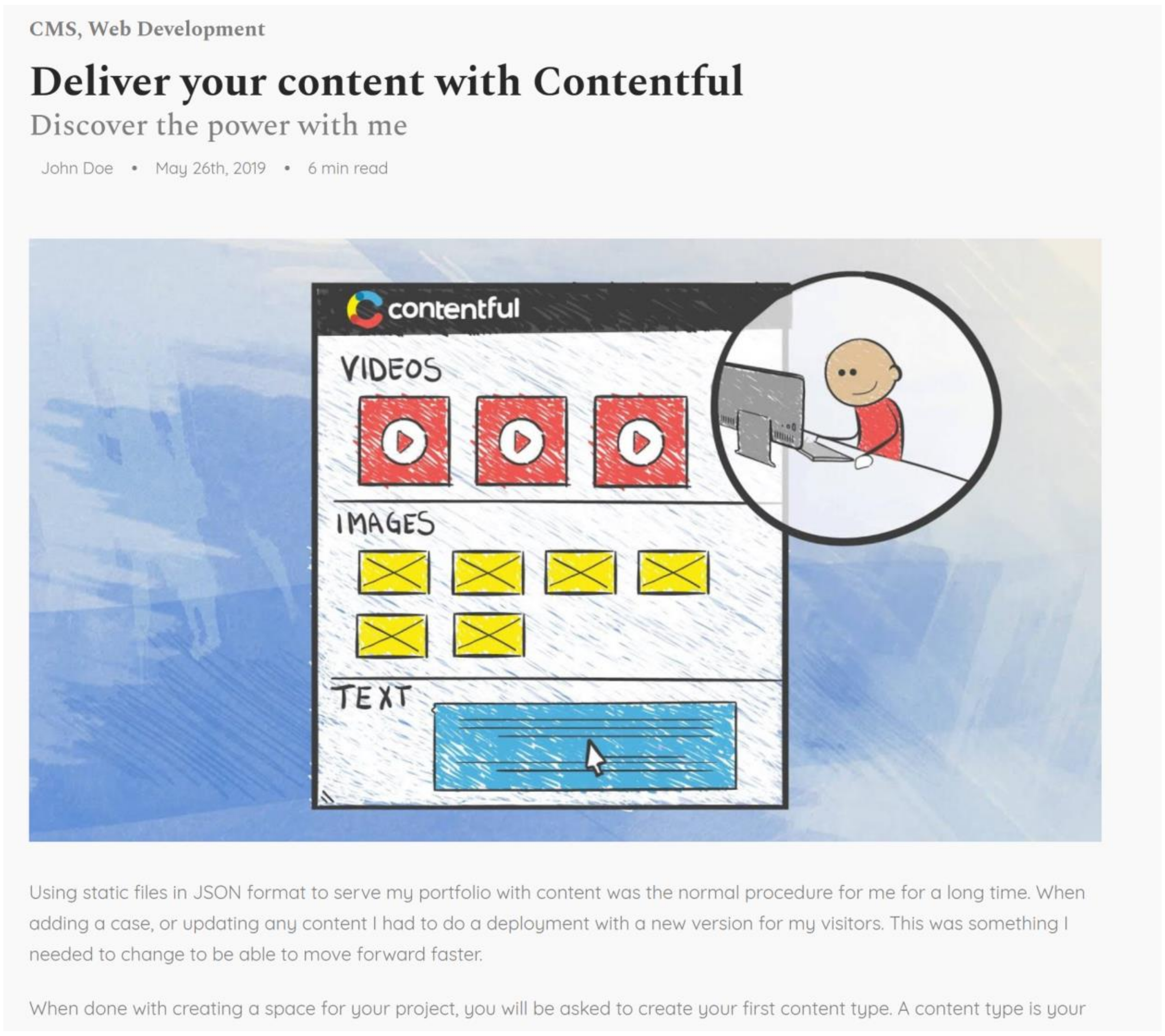
The coding

You can either create your own HTML/CSS from scratch, or you can use a little template we put together as a starting point. The template is here:

https://github.com/Majd-Darraj/HYF_contentful_workshop (you will need the "Index.html" and all the files in the "assets" folder.

If you keep them locally you should be able to just open the "index.html" directly in your browser – no need to set up a web server.

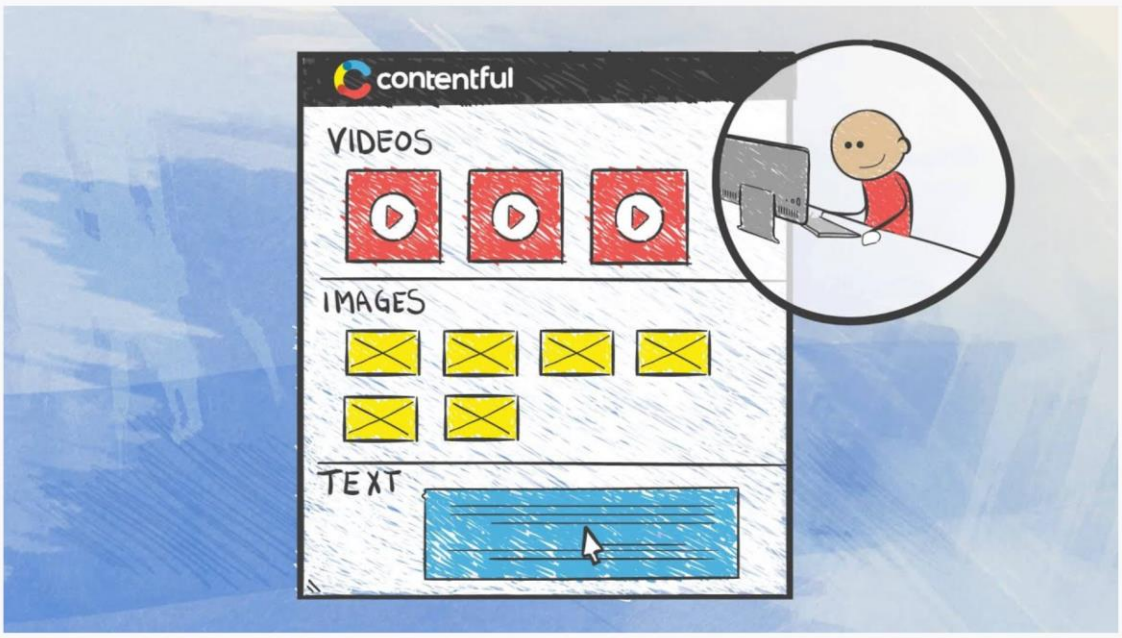
The template looks like this:


CMS, Web Development

Deliver your content with Contentful

Discover the power with me

John Doe • May 26th, 2019 • 6 min read



Using static files in JSON format to serve my portfolio with content was the normal procedure for me for a long time. When adding a case, or updating any content I had to do a deployment with a new version for my visitors. This was something I needed to change to be able to move forward faster.

When done with creating a space for your project, you will be asked to create your first content type. A content type is your

Notice the headline, subheader, image and mainbody? Now we want to make sure we manage the content in them from Contentful.

First off, we need a few javascript libraries, so before the </head> we will add this:

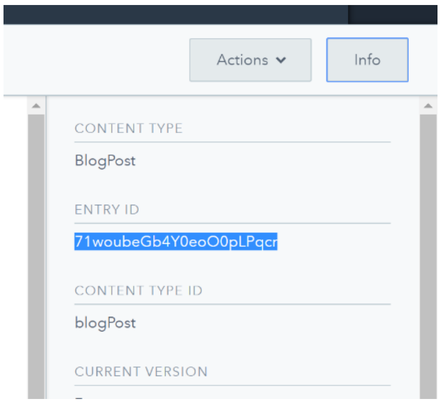
```
<script src="https://cdn.jsdelivr.net/npm/contentful@latest/dist/contentful.browser.min.js"></script>

<script>   var exports = {}; // quick fix because 'exports' is not defined in rich-text bundle below   </script>

<script src="https://unpkg.com/@contentful/rich-text-html-renderer@12.0.0/dist/rich-text-html-renderer.es5.js"
charset="utf-8"></script>
```

The first line is fetching the Contentful JS library, the 2 next is to fetch the Rich text renderer.

Next, we create our own <script> tags below the lines above and create a client to access managed content in our space by providing the proper space ID and access key:

```
var client = contentful.createClient({  space: '[spaceID]',  accessToken: '[accesstoken]'});
```

Next, we want to get the content item to show. We could do a search for it based on properties, but again – we'll stay simple. We'll use a hardcoded ID, but let is be possible to replace it by adding "?id=[new id]" in the url.



CONTENT TYPE
BlogPost

ENTRY ID
71woubeGb4Y0eoO0pLPqcr

CONTENT TYPE ID
blogPost

CURRENT VERSION

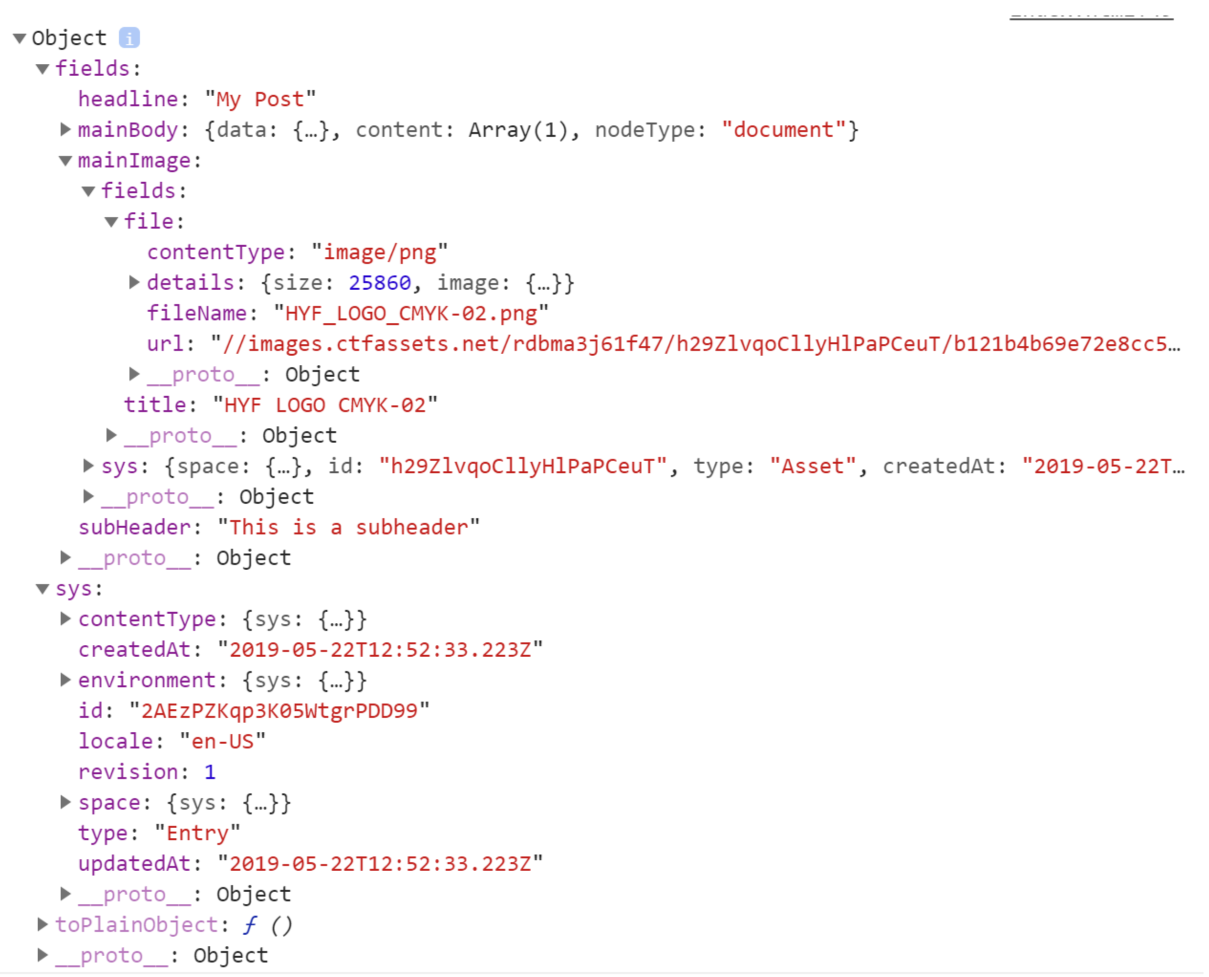
Note: You get the contentful ID of a content item in the "Info" box at the content item in Contentful.

```
var entryid="[my start page id]";

var urlParams = new URLSearchParams(location.search);

if(urlParams.has('id')){

            entryid=urlParams.get('id');

}
```

Now, that we have the entryid, it's straight forward to fetch the content from Contentful and simply replace the properties with it. Note that the entry fields are named using the field ID you saw when you created them in Contentful's content model.

```
client.getEntry(entryid)

  .then(entry => {

            console.log(entry);

            document.querySelector('#headline').innerHTML=entry.fields.headline;

            document.querySelector(‘#subheader’).innerHTML=entry.fields.subheader;

            document.querySelector('#mainImage').src="https://"+entry.fields.mainImage.fields.file.url;

            document.querySelector('#mainBody').innerHTML=documentToHtmlString(entry.fields.mainBody);


  });
```

Notice how we output the entry itself to the console? That's just for debugging purposes – this way you can see what the content object consists of if you look in your Developer Tools console in Chrome.



```
▼Object
  ▼fields:
      headline: "My Post"
    ▶mainBody: {data: {…}, content: Array(1), nodeType: "document"}
    ▼mainImage:
      ▼fields:
        ▼file:
            contentType: "image/png"
          ▶details: {size: 25860, image: {…}}
            fileName: "HYF_LOGO_CMYK-02.png"
            url: "//images.ctfassets.net/rdbma3j61f47/h29ZlvqoCllyHlPaPCeuT/b121b4b69e72e8cc5…
          ▶__proto__: Object
          title: "HYF LOGO CMYK-02"
        ▶__proto__: Object
      ▶sys: {space: {…}, id: "h29ZlvqoCllyHlPaPCeuT", type: "Asset", createdAt: "2019-05-22T…
      ▶__proto__: Object
      subHeader: "This is a subheader"
    ▶__proto__: Object
  ▼sys:
    ▶contentType: {sys: {…}}
      createdAt: "2019-05-22T12:52:33.223Z"
    ▶environment: {sys: {…}}
      id: "2AEzPZKqp3K05WtgrPDD99"
      locale: "en-US"
      revision: 1
    ▶space: {sys: {…}}
      type: "Entry"
      updatedAt: "2019-05-22T12:52:33.223Z"
    ▶__proto__: Object
  ▶toPlainObject: ƒ ()
  ▶__proto__: Object
```

Congratulations – you should now have a blog post file with managed content.

Note: This is purely for demonstration purposes. Often, you wouldn't rely on client-side javascript to load the content and complete the rendering as clients might have javascript turned off for security reasons. Instead, you can utilize NodeJS, .NET, PHP, Ruby or a similar server-side technology to render it – or maybe generate static pages based on managed content using a technology like GatsbyJS.

Additional exercises:

Here are some ideas for additional exercises you can try:

- Add fields in contentful and content manage the rest of the page – like the Tags (create a Text field, then check the box to make it a List), the author name, or the Date (the published date is already included in the entry object).
- Use the javascript API to retrieve a list of all BlogPost items and use that to produce a navigation menu on the pages.
- Create a StartPage type and a corresponding HTML that lists the blog posts by their title and links to them.