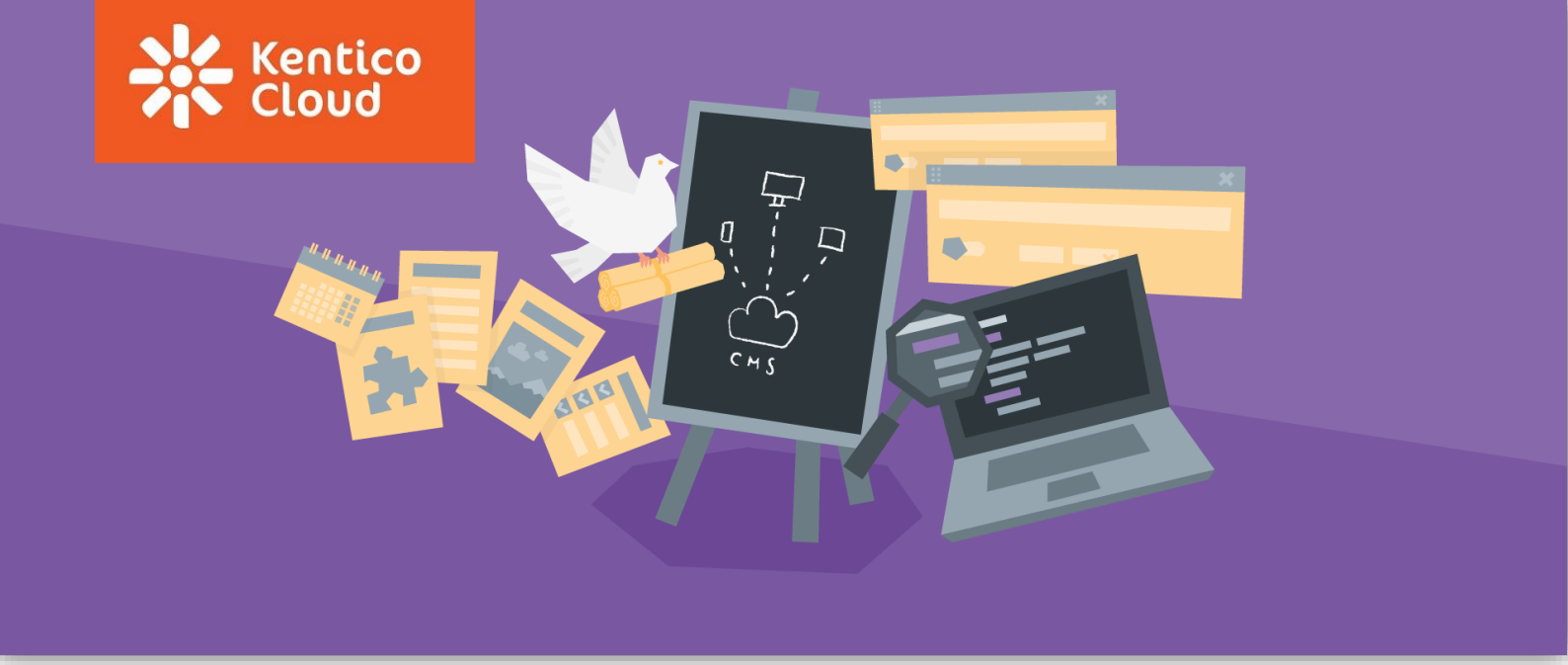
Kentico Cloud



Content modelling cheatsheet
&
Content repository audit checklist

This document provides a **collection of best practices & recommendations** to kick-off the content models for your project & avoid common mistakes.

Besides that you can also use it as a checklist of things to watch and evaluate during the content repository auditing activity. The checklist will be updated as the product evolved, or once we identify other common issues and bad practices.

Contents

General tips

Recommendations

- Think about the **resusability** all the time
- Try to keep the **number of content models low**
- Keep it **clear and simple** as much as possible for contributors, make it **intuitive**
- We suggest that if you don't want to give authors control over the styling, **simple text** is usually a better fit.
- Check if your content types have same **fields/attributes in common**.
  Would it be possible to take out those fields and replace them with a content type snippet?

Questions to ask for every model and elements

- What **platforms** will consume it?
- Is the **structure** going to be **reused** in any other content types?
- If there is a **list**, is it generated **dynamically or manually**?
- If there is a **list**, do you need to be able to **sort** the items **dynamically or manually**?
- Is the **element** going to be used for **filtering**?
- Are the **items** going to be **reusable**?

Content models, Content types

Naming conventions

Recommendations

- **Short and accurate names**
  Examples: Long description, Short description, Body copy, Excerpt

- **Names should not be channel specific**
  E.g. instead of "Description for mobile," it's better to use "Short description" and provide some limitations combined with additional information in guidelines. Thanks to that, "Short description" is more general and can be **reused** for other platforms

Common mistakes

- **Unclear names** – editors are confused and they don't know what content type to select when they need to create a specific peace of content

Guidelines

Recommendations

- Use **element guidelines** to provide additional information for content production, this might have a significant impact on the **efficiency** of work of content creators/editors. It **can save some time on both sides** by avoiding back and forth communication when something is unclear. Make yourself a favor, give people clear instructions, so they don't have to ask the same questions again. And if you see that people are asking again, that's a clear sign that guidelines are missing or can be improved.
- Use **general guidelines** to separate content sections and provide clear explanation for each of them, give people additional context e.g. about voice and tone
- It's a good practice to **separate content and metadata elements** by a guideline element

Common mistakes

- **Not using the guidelines at all**

Granularity and reusabilty

Recommendations

- Content should be modeled in **smaller chunks** so that it's easily **reusable** across project and multiple platforms. Reusability needs to be implemented based on the specific project requirements. There aren't any particular rules that can be applied in general.

Common mistakes

- **Duplicating content**
- Creating **channel or project occurrence specific** content variants

Structure

Recommendations

- **Avoid content blobs**. Content should be **structured** as much as possible

Common mistakes

- Creating structured content in rich text fields

Presentation independence

Recommendations

- Content models **should not contain elements**, that are supposed to keep information **related to the presentation** of the content
- Note: There are cases, when content authors want to control some presentation aspects of the produced content. Apart from semantic controls within richtext elements (e.g. headings, strong font), we don't have any other options how to provide such functionality. During the audits, we might at least observe how often users use specific elements to control the presentation and design solution according to the use cases and customer needs.

Common mistakes

- Creating content elements that are supposed to keep information about content presentation. E.g. image floating (right or left)

Metadata

Recommendations

- Elements for structured **metadata** should be provided **within the content models** (e.g. for personalization, customer journey specification, targeted platforms)
- Metadata should have **clear guidelines** to **explain their purpose**

Element settings (limitations, required fields, modular content limitations)

Recommendations

- **Wherever it makes sense, element limitations and restrictions should be provided** in order to guide content production process

Auto generating URL slugs

Recommendations

- **URL slugs** should be generated based on a predefined **model element**

Common mistakes

- Generating URL slug based on content item title

Taxonomy utilization

Recommendations

- Taxonomy should be used to **categorize content** and create **content repository ontology**
- Taxonomy **shouldn't be overly complex** so that it's **easy to navigate and use**
  E.g. maximum 3 levels deep and up to 15 terms per level

Common mistakes

- Using sitemap instead of taxonomy

Taxonomy vs. Multiple choice element

Recommendations

- You should use Taxnomy as a primary way to categorize your content. However, there are some differences that you should be aware regarding the Multiple choice elements
- Multiple choice element is flat structure while Taxonomy allows you to take advantage of hierarchy
- For Multiple choice element you can define some validation options
- Multiple choice element can be used and displayed in two different ways:
    1. As a list of checkboxes (0-N options can be selected)
    2. As a radio buttons (only 1 option can be selected
- Taxonomy is always displayed as a list of checkboxes
- If you need to use the value for filtering (via API and Taxonomy) than it's probably better to use taxonomy

Content type snippets

Recommendations

- Content type snippets should be used to **avoid repetition/duplicating of common group of content elements**

Custom elements

Recommendations

- You can create a custom elements for your specific needs. Your development team have to implement the logic that will power the custom element
  Exampes: Color picker, Map location, DAM integration

Content imported from external sources

Recommendations

- When you're putting together your content models, think also about the content that can be imported from external sources, e.g. old website, static files, Product Catalog, etc.
- You can then use the Content Management API to migrate your existing content into Kentico Cloud project, or update content in unpublished content items. Note, that the API cannot be used on published content items unless you create new versions of the items first.

Content items

Naming conventions

Recommendations

- **Short and accurate** names

Common mistakes

- **Prefixing** names with content type name or other repetitive text

Granurality (ruseability)

Recommendations

- Content should be produced in **smaller chunks** so that it's **easily reusable** across project and multiple platforms.

Richtext in-line modules vs. Linked content

Recommendations

- Whenever **only modular content is expected** within a particular element, **linked content** element should be used
- On the contrary, if **linked content** is supposed to be **combined** with some other content (e.g. text, assets), **rich text field is a good fit**

Common mistakes

- Using rich text field on places, where modular con, atent is more suitable

Building the Navigation - Navigation items

Recommendations

- **Navigation** should be done by using **navigation content items** (check the best practice in our documentation)

Hierarchical project structure (Sitemap)

Recommendations

- Be aware that Sitemap it‘s channel specific – Website only, you should rather think omnichannel to create future-proof conten.t, Even you’re building just a website now, it might change soon, as other channels are evolving pretty fast.
- But if you need, you can use Taxonomy to create a sitemap

Structuring, Linking and Embedding Content

When writing an editorial article or a blog post, or composing a landing page, usually you'd need to enrich your content with something more than just a text. You may want to include tweets, testimonials, or other predefined structures to make the content more appealing. Let's look at how you can achieve this by using components and content items in the rich text editor. On the other hand, Linked items elements enable you to link content together in your project, e.g. to link articles about a similar topic or to feature related products on an e-commerce site.

Components would be the first choice when it comes to including richer structures to your articles as they instantly become a natural part of the element. Alternatively, **content items** linked in the Rich text element can be helpful when placing content that will be reused in multiple places. Let's take a closer look at the benefits of both options.

how to compose your larger content out of single-use components. If you want to link reusable content items in different places across your project, you might be more interested in a guide on linking content together.

Components in the Rich text element

Consider using components in the following scenarios:

- You're writing an article and you'd like to include some additional structures to enrich your content, such as tweet, quote, or a video.
- You're composing a landing page containing feature blocks and product descriptions.

Both of these cases have one thing in common – you're writing content that is only going to be present in the context of the article or a landing page. And this is where components can come in handy.

Example:

- If you wish to include a tweet that's only relevant to the topic you're currently writing about, a component would be the way to go. By creating a **component** for the tweet, that tweet becomes a **natural part of the element**.
- In other words, you don't need to create a specific content item for something you **won't be using more than once**.

Reusing components

- If you later find some other possible use for the content in the component, you can always **convert component to the content item** by clicking the convert icon in the right corner. Note that this action cannot be undone.

Finding components in the project

- When creating a component in the Rich text element, the component will only exist in that particular content item. This means you won't find the component itself when searching for it in the Content. But there's really no need for it, as the component will be processed as part of the element.

Component workflow

- Components don't have their own workflow, they live and exist in the content item they were created in. If the parent content item gets published, so does the component. So if your content item contains only components, you don't need to worry about forgetting to review some of the content. Because component is a part of the Rich text element, it gets reviewed together with the element itself.

Content Items in the Rich text element

You can use content items linked in the Rich text element in the very scenarios you would use components, such as blog posts, or landing pages. The main difference here is that content items contained in the Rich text element are independent from the rest of the content. This means that they are accessible outside the parent item they're used in and thus ready to be reused in other content items as well.

Example:

- A customer testimonial that's placed in different parts of your web can also be reused in the article you're currently writing.

Reusing content items

- If you want to include a testimonial relevant to the topic you're writing about, it makes sense you would use a content item to do so. This way, you can re-use the item in the future by adding it to a different content item as well.

Linked Items

Use the Linked items element to build content from multiple content items. For example, it is suitable for the following scenarios:

- Creating larger content items composed of smaller content items. For example, you can build landing pages by repeating content items.
- Relating content that belongs together. For example, you have an author’s bio that you want to include at the end of the individual blog posts.

Recommendations

- Restrict which content type/s (one or multiple) can be used as Linked Content for each element

Common mistakes

- Adding multiple Linked Content elements to your content model for each individual linked piece of content instead of adding multiple items into one Linked Content element

Assets

Create content types for desired asset types

If you need to provide some additional information for your assets you may consider creating a speial content type(s) for assets which will then allow you to add more elements/fields (e.g. description, categorization, ...)

Recommendations

- Organize assets into folders

Based on the complexity of the project, content strategists can either:

- Create special "Asset" content type to store assets with additional metadata
- Create **specialized content types** like "Image", "Video" or "E-book", if you need to store different additional metadata with them

Copywriters, then, can include these assets in an article as **Linked items or Components**.

Example:

Image (Content type)

Consisting of the following elements:

- Image (Asset)
- Image description (Text)
- Asset Category (Taxonomy)
- License agreement (Multiple choice)
- Author (Linked items)

Benefits

This method offers a versatile and highly customizable way to organize assets:

- With Taxonomy, you can **categorize your assets** (to find them later)
- With any element, you can **include additional metadata** to your assets
- Within the Linked item, you can **replace the asset** with a newer version
- With multiple asset elements within a content type, you could store **multiple stylistic variants** of the image

Image descriptions

Recommendations

- **Provide asset descriptions** as they are valuable metadata describing asset contents

Bonus recommendations

Using preview URLs

Recommendations

- Use preview functionality for better content visualization and authoring experience

Workflow states

Recommendations

- Workflow states should reflect content authoring process and organization structure
- Workflow steps should have short and apt name describing their purpose

Language variants

Recommendations

- Language variants should should have fallbacks defined to provide default variant

Resources

- Kentico Cloud Docs
- Developer Hub

- Define content structure
- Compose and link content
- Managing navigation menus
- Content migration – Importing to KC