# Extend the content editing features with custom elements

March 17, 2022 • David Klement • 5 min read

You can extend capabilities of Kontent by implementing additional features you need for your project, such as a 3rd-party video service. These extensions are called custom elements. Custom elements help you extend the built-in functionality of Kontent by Kentico and take the content editing experience to another level.

**Key points**

- A custom element is a small HTML app that acts as an extension of Kontent's native capabilities.
- To use your extension in a Kontent project, add a custom element into a content type and specify the URL of your extension.
- You need to securely host your custom elements yourself.

## What is a custom element?

Custom elements are small HTML apps that exist in a sandboxed within the Kontent app and interact with Kontent through the Custom Elements API. They enable you to extend the set of default content type elements to include the functionality you need.

Have a look at a few examples of extending Kontent capabilities using custom elements:

- Enable selecting assets from a 3rd-party digital asset management system (DAM), such as Bynder.
- Add an AI-powered content personalization system like Recombee.
- Let your content creators write the way they're used to by implementing a custom WYSIWYG text editor.

## Gallery of custom elements

To see all the custom element samples, check out our gallery of custom elements. There, you can choose whatever suits your needs and add it to Kontent.

We're using the Markdown editor for example purposes in this article. You can apply the same principles to add any custom element from the gallery.

If you don't find in the gallery what you need, you can always create your own custom element.

## Host your custom element securely

Custom elements are HTML applications loaded in an . They need to be hosted so the browser can fetch the file. You need to host your custom element yourself or use a service like Netlify.

- If you decide to host your custom element on your own servers, you need to
  - setup the appropriate domain certificate for it;

- use the (or ) header to specify an allowed origin, in this case, . For more details, see the MDN Web Docs.
- To test your custom element locally, we suggest to use a secure tunnel to your localhost. You can use a service like ngrok that connects to your local application and provides a secure public URL for access from the Kontent app.

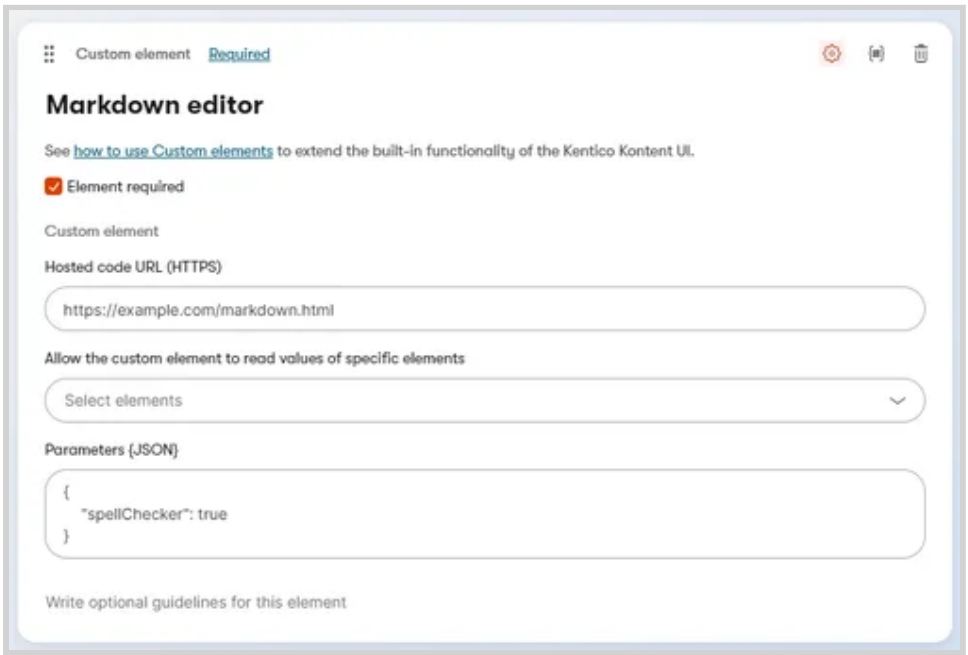
Your extension is separated from the rest of Kontent through the attribute on the . The following sandbox flags are enabled:

- allow-forms – allows form submission
- allow-scripts – allows JavaScript to run inside the
- allow-popups – allows popups
- allow-same-origin – allows the document to maintain its origin
- allow-modals – allows modal dialogs

## Add the custom element to your project

To have your custom element appear in content items, you need to add it to the content type of the items.

1. In **Content model**, open a content type for editing.
2. Click *Custom element* on the right to add a new custom element.
3. Give it a name.
4. Paste the URL where you host your custom element into **Hosted code URL (HTTPS)**.
5. If your custom element has some configuration parameters, set them in **Parameters {JSON}**.
6. Click **Save changes**.




If you set a custom element as required, Kontent checks if the element contains any value. Any other value than is considered valid and won't prevent an item from getting published.

### JSON parameters

The JSON parameters enable you to configure your element and reuse it in various contexts. In some cases, you may want different settings for different projects, content types, or even elements.

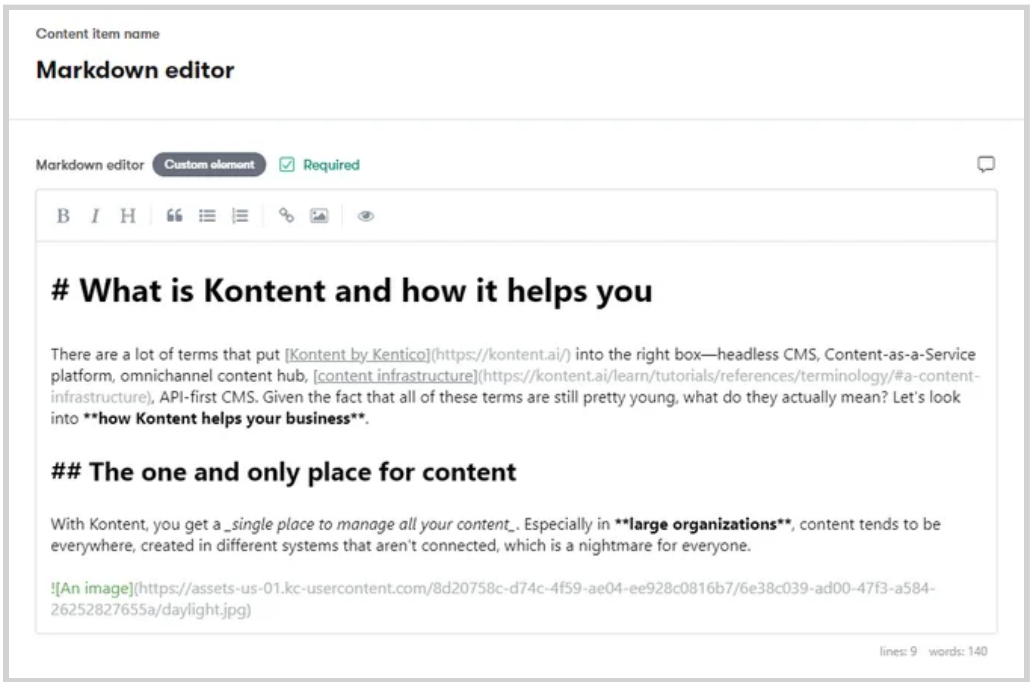
For instance, in the case of the Markdown editor, you could configure yours to accept JSON parameters for activating and deactivating spellcheck and tools like inserting images. Then, you would enable spellcheck or images only in places where it makes sense.

You could also change the custom editor's HTML so that all of the tools in it can be activated and deactivated via JSON parameters. That way, each custom element could have a different toolset.

> ⚠ Never store your API keys or any other sensitive data in the JSON parameters field. It's not secure! Use server-side proxy to store sensitive data.

## See the final result

After you're done setting up the content type, you can use the custom element inside content items. If you implement the Markdown editor, it looks like this:




When you fetch this content item via the Delivery API, you'll get your markdown in the `value` of the custom element, similar to the following.

JSON

## What's next?

- Check out the Custom Elements API specification to see all the available API methods.
- See what integrations we offer, where custom elements typically play a crucial role.
- Protect your sensitive data in custom elements.