Use webhooks for automatic updates

March 28, 2023 • Jan Cerman • 10 min read • .NET

Webhooks allow you to integrate Kontent.ai with other apps and automate your processes. Think of webhooks as programmatic notifications that let your app know when something changes.

What can you do with webhooks?

Here are a few example scenarios:

- Update the search index with your content using services like Algolia, Azure Search Index, and more.
- Trigger a new build process and redeploy your application.
- Notify your team by sending an email, posting a message to a Slack channel, or moving a card inside Trello.
- Schedule a social media post featuring the newly published content item.
- Invalidate your app's cache to make sure users see the latest content.

You can also react to workflow changes in your content items to:

- Automatically send content for translation and receive notifications when the translation is done.
- Notify reviewers that the content is ready for review.

How webhooks work in a nutshell

1. Kontent.ai sends an HTTP POST request to the URL you specify right after something changes in your project.
   - If you don't have such a URL or can't react at any time, consider using Sync API.
2. The payload of the POST request is a JSON with information about the affected content items and the type of change.
3. The header of the request contains a secret token you can use to verify the message.
4. Your application responds to Kontent.ai and processes the request. For example, you can go through the payload and get fresh content.
5. If your application is unavailable or fails to process the request, the request is sent again as per the retry policy.

Let's go through the process in more detail.

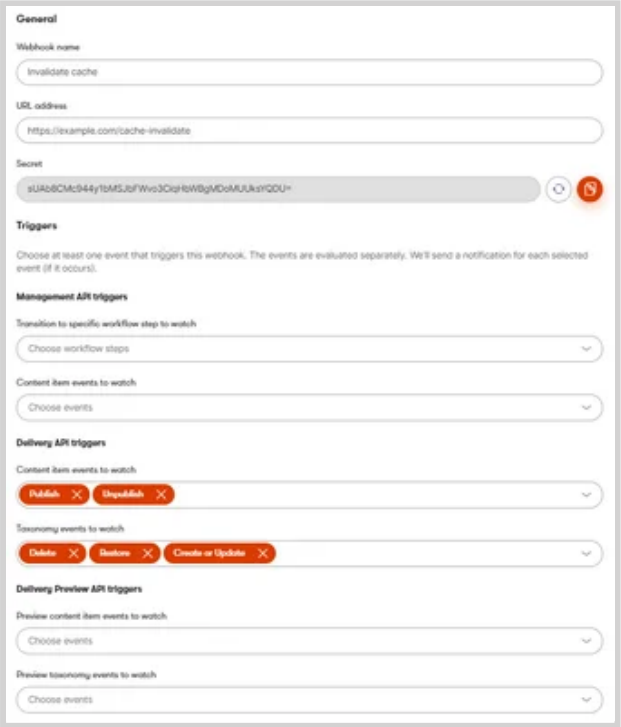
Create a webhook

To register a new webhook:

1. In Kontent.ai, go to **Environment settings** > **Webhooks**.

2. Click **Create new webhook**.
3. Enter a name for the webhook such as *Invalidate cache*.
4. Enter a publicly available URL address of your app's endpoint, such as `https://example.com/cache-invalidate`.
5. (Optional) Choose the events to trigger the webhook. By default, the events related to changes in content items and taxonomies are selected.
6. Click **Save**.

Whenever the content events are triggered, Kontent.ai sends a notification to your endpoint.



Creating a webhook in Kontent.ai

Validate received notifications

Kontent.ai sends notifications (as POST requests) to the webhook URL you specified. Receiving your notifications might take a few minutes or possibly longer. The notifications may sometimes come in batches because the content changes are processed dynamically based on load.

The notifications sent to your webhook URL come with the `X-KC-Signature` header. This header contains a checksum of the JSON payload that you can use to verify the authenticity of the notification. The signature is a base64 encoded hash generated using an HMAC-SHA-256 with a secret key.

To verify the received notification, use the JSON payload (that is the raw body of the notification as-is) and the generated webhook *Secret* as the secret for the HMAC function. The secret is unique for each webhook. Once you generate a hash, compare the hash value with the `X-KC-Signature` header value.

C#

```
// Tip: Find more about .NET SDKs at https://kontent.ai/learn/net
using System;
using System.Security.Cryptography;
using System.Text;

// Example of generating the hash to verify the notification
private static string GenerateHash(string message, string secret)
{
    secret = secret ?? "";
    UTF8Encoding SafeUTF8 = new UTF8Encoding(encoderShouldEmitUTF8Identifier: false,
throwOnInvalidBytes: true);
    byte[] keyBytes = SafeUTF8.GetBytes(secret);
    byte[] messageBytes = SafeUTF8.GetBytes(message);
    using (HMACSHA256 hmacsha256 = new HMACSHA256(keyBytes))
    {
        byte[] hashmessage = hmacsha256.ComputeHash(messageBytes);
        return Convert.ToBase64String(hashmessage);
    }
}
```

Get the latest content

After you get a notification about changed content, you might want to explicitly request the changed content from the Delivery API. To do this, send a standard request to Delivery API with the `X-KC-Wait-For-Loading-New-Content` header set to `true`.

If the requested content has changed since the last request, the header determines whether to wait while fetching content. By default, when the header is not set, the API serves stale content (if cached by the CDN) while it's fetching the latest content to minimize wait time.

C#

```
// Tip: Find more about .NET SDKs at https://kontent.ai/learn/net
using Kontent.Ai.Delivery;

// Initializes a client that retrieves the latest version of published content
IDeliveryClient client = DeliveryClientBuilder
    .WithOptions(builder => builder
        .WithProjectId("<YOUR_PROJECT_ID>")
        .UseProductionApi
        .WaitForLoadingNewContent
        .Build())
    .Build();

// Gets a content item
// Create strongly typed models according to https://kontent.ai/learn/net-strong-types
IDeliveryItemResponse<Article> response = await client.GetItemAsync<Article>
("my_article");
```

```
15
Article item = response.Item;
```

Retry policy

If your application responds with a `20X` HTTP status code, the notification delivery is considered successful. Any other status code or a request timeout (which occurs after 60 seconds) will result in a repeated notification delivery.

On the first unsuccessful delivery, Kontent.ai tries to send the notification again in 1 minute. If the delivery is unsuccessful, the delay between resending the notification increases exponentially to a maximum of 1 hour. The specific delay intervals are (in minutes): 1, 2, 4, 8, 16, 32, and 60. When the delay reaches 60 minutes, Kontent.ai tries to deliver the notification every hour for up to 3 days, after which the notification is removed from the queue.

All notifications are delivered in the order they were created. For example, if a notification is successfully delivered after 4 minutes, the notifications created after it will follow in the original order.

Email notifications

If there's a problem with webhook delivery, users with the *Manage APIs* permission get an email in these cases:

- Notification delivery is repeatedly failing for 1 hour. This email is sent only once for each registered webhook.
- Notification delivery is repeatedly failing for 3 days. After 3 days, the notification won't be delivered again.
- Notification delivery was successful after several failed attempts. This email is only sent if you previously received an email notification about a failed delivery.

Debug webhooks

If you get an email about a failing webhook, you can find details in Kontent.ai > **Environment settings** > **Webhooks**.

For an overview of the health of your webhooks, each webhook in your list has a colored **health status** next to its name.

- Light grey – Ready for message. This appears for newly created webhooks before any change to published content has been made (so no notification has been sent).
- Green – Working. This appears for webhooks that have properly delivered notifications.
- Red – Failing. This appears for webhooks that have not been delivered properly (received a response other than a `20X` HTTP status code). These webhook notifications are still being sent based on the retry policy. You can also reset the webhook if you've issued a fix and want to reset the retry policy.
- Grey – Dead. This appears for webhooks where delivery has repeatedly failed and no notifications have been accepted for 7 days. After this time, no more notifications will be

sent and all notifications waiting to be sent will be deleted. You can revive a dead webhook by resetting it.

Reset and revive webhooks

If your webhook is failing or dead, you can reset it. Resetting a webhook means that Kontent.ai attempts to resend the last failed notification, effectively resetting the webhook's retry policy.

1. In Kontent.ai, go to **Environment settings** > **Webhooks**.
2. Select a failing or dead webhook to open its details.
3. Click **Reset**.

Disable and enable webhooks

If you plan to make changes to your content model, you might want to disable your webhooks first and then adjust your app to the new content model. Disabling webhooks also helps if you plan to release hundreds or thousands of content items in a short amount of time.

When you disable a webhook, the webhook stops sending notifications and deletes any unsent notifications. Disabled webhooks ignore their triggers and don't create any new notifications.

1. In Kontent.ai, go to **Environment settings** > **Webhooks**.
2. Click a webhook to open its details.
3. Click **Disable**.

After you enable the webhook, the webhook is ready to send new notifications whenever content changes.

Get more information about your webhooks

For more information about each webhook, click . You'll find a list of all notifications from the last 3 days, sorted from newest to oldest.

You can filter the list to show everything, only failures (at any time in sending the message), or only active failures (where the last response was a failure). Click **Refresh** to reload the list.

Each notification in the list shows:

- The number of times the delivery has been attempted.
- A button </> to view the most recent response.
  - In the window that pops up, a button to copy the response to the clipboard.
- The date and time when the most recent delivery attempt was made.

What's next?

- See how to work with webhooks hands-on in our intro course.
- Dive deeper into the structure of webhooks and their triggers.
- React to changes in your content at your own pace with Sync API.
- Understand how to add search to your application.
- Need help with integrating webhooks? Ask the community in Stack Overflow.