Available webhook events

Last Modified on 07/29/2025 4:30 pm EDT

Below are the knowledge base events you can trigger an email notification on, along with some sample email notifications.

Need something else?
If you have an event that you'd like to receive email notifications on that's not listed below, contact us to request we add it!

Article created

The **Article created** event is triggered when an author selects **Create** for a new article.

Sample Article created webhook payload

The Article created webhook sends type `article.create`, containing `data` with the `article` and the details for the author (`user`) who created it:


```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.create",
  "created": {{webhook firing UNIX timestamp}}
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": draft,
      "callout_expire": null
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    }
  }
}
```

Article updated

The **Article updated** event is triggered whenever an author saves changes to an article.

It does not provide information about what was changed.

Sample Article updated webhook payload

The Article updated webhook sends type `article.update`, containing `data` with the `article`, the details for the author (`user`) who saved the change, and a link to the live article:


```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.update",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": {{article publishing status, e.g. "published"}},
      "callout_expire": {{callout expiration date, UNIX timestamp, if it exists}}
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    },
    "live_article_link": {{link to article in live knowledge base}}
  }
}
```


Article updated callout added

The **Article updated callout added** event is triggered when an author saves an article with the Updated callout. It's only triggered if the article previously didn't have an existing New/Updated callout.

Sample Article updated callout added webhook payload

The Article updated callout added webhook sends type `article.updatedCallout`, containing `data` with the `article`, the details for the author (`user`) who added the callout, and a link to the live article:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.updatedCallout",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": "published",
      "callout_expire": {{callout expiration date, UNIX timestamp}}
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    },
    "live_article_link": {{link to article in live knowledge base}}
  }
}
```


Article new callout added

The **Article new callout added** event is triggered when an author saves an article with the New callout. It's only triggered if the article previously didn't have an existing New/Updated callout.

Sample Article new callout added webhook payload

The Article new callout added webhook sends type `article.newCallout`, containing `data` with the `article`, the details for the author (`user`) who added the callout, and a link to the live article:

```
{
 "sender": {
  "application": "KnowledgeOwl",
  "webhook_id": {{webhook id}},
  "token": {{token}}
 },
 "type": "article.newCallout",
 "created": {{webhook firing UNIX timestamp}},
 "data": {
  "article": {
   "id": {{article id}},
   "name": {{article title}},
   "status": "published",
   "callout_expire": {{callout expiration date, UNIX timestamp}}
  },
  "user": {
   "id": {{author id}},
   "email": {{author email address}},
   "first_name": {{author first name}},
   "last_name": {{author last name}}
  },
  "live_article_link": {{link to article in live knowledge base}}
 }
}
```


Article published

The **Article published** event is triggered when when an author publishes a previously unpublished article, either in the article editor or through a bulk edit in Manage.

Sample Article published webhook payload

The Article published webhook sends type `article.publish`, containing `data` with the `article`, the details for the author (`user`) who published the article, and a link to the live article:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.publish",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": "published",
      "callout_expire": null
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    },
    "live_article_link": {{link to article in live knowledge base}}
  }
}
```


Article archived

The **Article archived** event is triggered when an author saves an article with the Archived publishing status or archives articles in Manage.

Sample Article archived webhook payload

The Article archived webhook sends type `article.archive`, containing `data` with the `article` and the details for the author (`user`) who archived the article:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.archive",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": "archived",
      "callout_expire": null
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    }
  }
}
```


Article deleted

The **Article deleted** event is triggered when an author deletes an article, whether from the Articles page, the article editor, or as a bulk edit in Manage.

Sample Article deleted webhook payload

The Article deleted webhook sends type `article.delete`, containing `data` with the `article` and the details for the author (`user`) who deleted the article:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.delete",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": "deleted",
      "callout_expire": null
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    }
  }
}
```


Article publishing status changed

The **Article publishing status changed** event is triggered when when an article's saved with a different publishing status.

This event will also trigger any time the Article published, Article archived, and Article deleted events trigger.

Sample Article publishing status changed webhook payload

The Article publishing status changed webhook sends type `article.statusChange`, containing `data` with the `article`, the details for the author (`user`) who changed the article's status, and a link to the live article:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "article.statusChange",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "article": {
      "id": {{article id}},
      "name": {{article title}},
      "status": {{publishing status}},
      "callout_expire": null,
      "prior_status": {{previous publishing status, e.g. "draft"}},
      "new_status": {{previous publishing status, e.g. "published"}}
    },
    "user": {
      "id": {{author id}},
      "email": {{author email address}},
      "first_name": {{author first name}},
      "last_name": {{author last name}}
    },
    "live_article_link": {{link to article in live knowledge base}}
  }
}
```


Comment created

The **Comment created** event is triggered when a reader submits a new comment or an author creates a new comment from **Reporting > Comments.**

Sample Comment created webhook payload

The Comment created webhook sends type `comment.create`, containing `data` for the `comment`, the API link for the comment (`api`), and the details of the article or category the comment was left on:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "comment.create",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "comment": {
      "id": {{comment id}},
      "content": {{full text of comment}},
      "poster": {{Name and email of commenter, e.g. "Linus Owl (linus@knowledgeowl.com)"}},
      "status": "pending",
      "article_id": {{article or category id}}
    },
    "api": "https://app.knowledgeowl.com/api/head/comment/{{comment id}}.json",
    "article_name": {{article or category title}},
    "article_edit_link": {{link to article or category in editor}},
    "article_type": {{"Article" for article; "Category" for category}}
  }
}
```


Comment deleted

The **Comment deleted** event is triggered when an author deletes a comment.

Sample Comment deleted webhook payload

The Comment deleted webhook sends type `comment.delete`, containing `data` with the `comment`, the API endpoint for the comment (`api`), the details of the article or category the comment was left on, and the details for the author (`user`) who deleted the comment:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "comment.delete",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "comment": {
      "id": {{comment id}},
      "content": {{full text of comment}},
      "poster": {{Name and email of commenter, e.g. "Linus Owl (linus@knowledgeowl.com)"}},
      "status": "deleted",
      "article_id": {{article or category id}}
    },
    "api": "https://app.knowledgeowl.com/api/head/comment/{{comment id}}.json",
    "article_name": {{article or category title}},
    "article_edit_link": {{link to article or category in editor}},
    "article_type": {{"Article" for article; "Category" for category}}
    "user": {
      "id": {{author id}},
      "first_name": {{author first name}},
      "last_name": {{author last name}},
      "email": {{author email address}},
      "icon": {{author icon URL}}
    }
  }
}
```

Comment updated

The **Comment updated** event is triggered whenever an author edits a comment, such as by approving or deleting it.

Sample Comment updated webhook payload

The Comment updated webhook sends type `comment.update`, containing `data` with the `comment`, the API endpoint for the comment (`api`), the details of the article or category the comment was left on, and the details for the author (`user`) who updated the comment:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "comment.update",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "comment": {
      "id": {{comment id}},
      "content": {{full text of comment}},
      "poster": {{Name and email of commenter, e.g. "Linus Owl (linus@knowledgeowl.com)"}},
      "status": {{comment status: "pending", "deleted", or "approved"}},
      "article_id": {{article or category id}}
    },
    "api": "https://app.knowledgeowl.com/api/head/comment/{{comment id}}.json",
    "article_name": {{article or category title}},
    "article_edit_link": {{link to article or category in editor}},
    "article_type": {{"Article" for article; "Category" for category}}
    "user": {
      "id": {{author id}},
      "first_name": {{author first name}},
      "last_name": {{author last name}},
      "email": {{author email address}},
      "icon": {{author icon URL}}
    }
  }
}
```


Comment status changed

The **Comment status changed** event is triggered when an author changes a comment's status, for example from Pending to Approved or Deleted.

Sample Comment status changed webhook payload

The Comment status changed webhook sends type `comment.statusChange`, containing `data` with the `comment`, the API endpoint for the comment (`api`), the details of the article or category the comment was left on, and the details for the author (`user`) who changed the comment's status:

```
{
  "sender": {
    "application": "KnowledgeOwl",
    "webhook_id": {{webhook id}},
    "token": {{token}}
  },
  "type": "comment.statusChange",
  "created": {{webhook firing UNIX timestamp}},
  "data": {
    "comment": {
      "id": {{comment id}},
      "content": {{full text of comment}},
      "poster": {{Name and email of commenter, e.g. "Linus Owl (linus@knowledgeowl.com)"}},
      "status": {{comment status: "pending", "deleted", or "approved"}},
      "article_id": {{article or category id}}
      "previous_status": {{previous comment status, e.g. "pending"}},
      "new_status": {{new comment status, e.g. "approved"}}
    },
    "api": "https://app.knowledgeowl.com/api/head/comment/{{comment id}}.json",
    "article_name": {{article or category title}}
    "article_edit_link": {{link to article or category in editor}},
    "article_type": {{"Article" for article; "Category" for category}}
    "user": {
      "id": {{author id}},
      "first_name": {{author first name}},
      "last_name": {{author last name}},
      "email": {{author email address}},
      "icon": {{author icon URL}}
    }
  }
}
```


Contact form submitted

The **Contact form submitted** event is triggered when a reader submits the Contact Form.

Sample Contact Form submitted webhook payload

The Contact Form submitted webhook sends type `contactForm.submit`, containing `data` with the `ticket_fields` and `submission` details:

```
{
 "sender": {
   "application": "KnowledgeOwl",
   "webhook_id": {{webhook id}},
   "token": {{token}}
 },
 "type": "contactForm.submit",
 "created": {{webhook firing UNIX timestamp}},
 "data": {
   "ticket_fields": {
     "subject": {{contact form submission subject}},
     "content": {{contact form submission body}},
     "from_email": {{submitter's email address, e.g. "linus@knowledgeowl.com"}},
     "from_name": {{submitter's name, e.g. "Linus Owl"}},
     "ip_address": "00.00.00.000",
     "browser": "Chrome",
     "OS": "Windows",
     "ua": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36"
   },
   "submission_time": {{contact form submission UNIX timestamp}},
   "submission_status": "success",
   "subject": {{contact form submission subject}},
   "body": {{contact form submission body}},
   "sender_name": {{submitter's name, e.g. "Linus Owl"}},
   "sender_email": {{submitter's email address, e.g. "linus@knowledgeowl.com"}}
 }
}
```

Reader metadata

If you follow the instructions in What data is collected in the Contact Form? to disable storing metadata, the `ticket_fields` won't be included, but the remaining data fields will be.