
# API Technical Guide: Content Block

**Cheetah Messaging**

# Table of Contents

# 1 Introduction

## Purpose

The purpose of this document is to provide an overview of the CONTENT BLOCK API endpoint within the Cheetah Messaging platform. This document discusses the intended use of the CONTENT BLOCK endpoint, and provides technical details for how to implement the endpoint.

## Overview

The CONTENT BLOCK endpoint is used to create, view, edit, and delete Content Blocks. Content Blocks are "containers" of message content that can be reused across Campaigns. If you repeat similar content across Campaigns, then setting up that material within a Content Block can save you the time of having to enter and re-enter that content in each separate Campaign. You can instead define that content once in a Content Block, then simply assign that Content Block to each Campaign that needs it.

This endpoint requires authentication using OAuth 2.0, and supports JSON and XML messages.

The URLs for this endpoint are:

- **North America**:
- https://api.eccmp.com/services2/api/ContentBlock
- https://api.eccmp.com/services2/api/ContentBlock/{id}/RelatedCampaigns
- https://api.eccmp.com/services2/api/ContentBlock/{id}/PickUpChanges
- **Europe**:
- https://api.ccmp.eu/services2/api/ContentBlock

- https://api.ccmp.eu/services2/api/ContentBlock/{id}/RelatedCampaigns
- https://api.ccmp.eu/services2/api/ContentBlock/{id}/PickUpChanges
    - **Japan**:
- https://api.marketingsuite.jp/services2/api/ContentBlock
- https://api.marketingsuite.jp/services2/{id}/api/RelatedCampaigns
- https://api.marketingsuite.jp/services2/{id}/api/PickUpChanges

## Methods

The CONTENT BLOCK endpoint supports the following HTTP methods:

- **POST**: Create a new Content Block.
- **GET**: Retrieve information about a specified Content Block.
- **PUT**: Submit modifications to an existing Content Block.
- **DELETE**: Delete a specified Content Block.
- **POST**: Convert HTML content to plain text.
- **POST**: Convert Cascading Style Sheet (CSS) styles to inline styles.
- **GET**: Retrieve a list of all Campaigns that use a specified Content Block.
- **GET**: Retrieve a list of all Campaigns that use a specified Content Block, and that have unapplied changes in a launched Campaign.
- **PUT**: Execute the Pick Up Changes process for all Campaigns that use a specified Content Block, and that have unapplied changes in a launched Campaign.

## Authentication

Access to the CONTENT BLOCK endpoint requires that you first be authenticated within the platform. Within Messaging, authentication is handled by OAuth 2.0. To authenticate with OAuth 2.0, you must first obtain a "Consumer Key" and a "Consumer Secret." Both of these

values are managed at the user level, and can be obtained from within the Messaging application.

Next, you'll use your Consumer Key and Consumer Secret to request a "token." A token is a text string that, when provided in a request message, will allow the user access to the requested service. Tokens are valid only for a certain period of time.

For more details on how to authenticate your API request, please see the *Messaging: API How-to Guide*.

# 2 Create a Content Block

## Overview

This section describes how to create a new Content Block via a POST request to the CONTENT BLOCK endpoint.

## Parameters

The options and parameters described in this section explain how to create a new Content Block consisting of one or more "format versions." Most channels in Messaging allow you to define multiple format versions of the message content in order to accommodate the different devices and applications used by recipients to view your message. A common example is an Email Campaign that includes both an HTML version and a Plain Text version. If a recipient has an email application that's not configured to view HTML messages, he or she can still see the Plain Text version of your message.

### entity_id

This integer parameter is required.

The **entity_id** parameter represents the **Entity ID** of the Content Block's source table.

Example:

```
"entity_id": 100
```

### cust_id

This integer parameter is required.

The **cust_id** parameter represents the Customer ID of your Messaging account. The Customer ID is a unique, system-generated identifier for every Messaging client account. This value isn't displayed anywhere within the Messaging application, so you must retrieve it by means of an API request (several different endpoints will return the

Customer ID as part of the response message), or speak to your Client Services Representative, who can provide you with this value.

Example:

```
"cust_id": 394
```

### type_id

This string parameter is required.

For a Content Block, the value of this parameter must be "PARAGRAPH."

Example:

```
"type_id": "PARAGRAPH"
```

### contBodies

This object specifies the format versions for your Content Block, and the message content for each format version. Within this same object, you may have one or more format versions (HTML, Plain Text, etc.), optionally with different content for each version.

Example:

```
"contBodies":
 [
  {
    "type_id": "HTML",
    "usage_mask": "EMAIL",
    "body": "<html> <body> Hello {(first_name)}!<br/><br/> Please
note that this is test HTML Content.<br/> {[Optout|46046]}</body>
</html>"
  },
  {
    "type_id": "TEXT",
    "usage_mask": "EMAIL",
    "body": "Hello {(first_name)}! Please note that this is test
Plain Text Content. {[Optout|46046]}"
  }
 ]
```

The parameters in this object are described below in more detail.

#### type_id

This string parameter is required.

The **type_id** parameter is used in combination with the **usage_mask** parameter (described below) to define the format version of the content.

The possible values for this parameter are:

- HTML
- TEXT
- XML

Example:

```
"type_id": "HTML"
```

### usage_mask

This string parameter is required.

The **usage_mask** parameter is used in combination with the **type_id** described above to define the format version of the content.

Most of the format versions in Messaging are actually groups containing multiple sub-options. For example, the "HTML" format version contains sub-options for: Email Message, Web, Viral, Share to Social, Mobile Web, iPhone, and Keitai Mail. By default, all of these sub-options are contained within the parent HTML version, meaning that the same HTML content will be used for all of those different contexts.

The **usage_mask** parameter allows you to pull one or more of those sub-options out of the parent version, and create a new, separate format version. This process is referred to as "promoting" a format option into a new group. For example, if you need the "Mobile Web" version of your HTML content to be different than the "Email Message" HTML content, you could promote "Mobile Web" to its own format version, and then provide the content unique to "Mobile Web" recipients.

This parameter can optionally contain multiple values, separated by commas.

Example:

```
"usage_mask": "EMAIL, WEB_MOBILE"
```

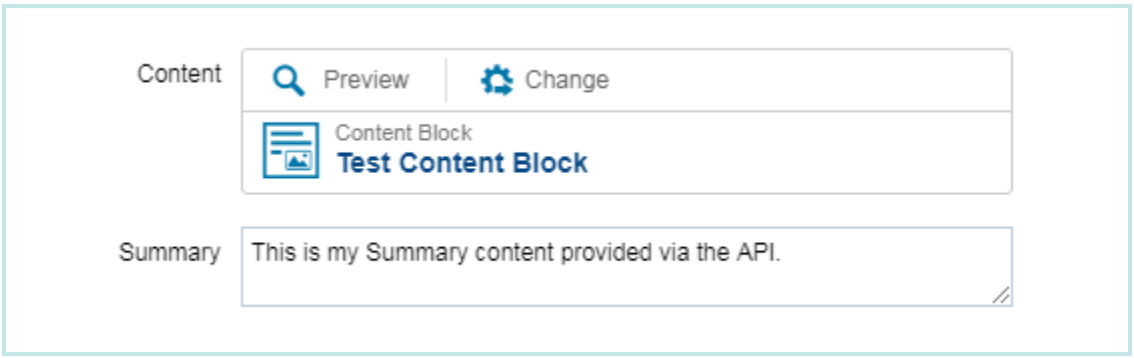
The valid values and combinations with **type_id**, along with the name used within the application's user interface, are listed below.

Some of the **usage_mask** values described below are designed as "bundles" of commonly used format versions.

| type_id | usage_mask | User Interface Name |
|---|---|---|
| HTML | EMAIL | HTML > Email Message |
| HTML | WEB_IPHONE | HTML > iPhone |
| HTML | WEB_SOCIAL | HTML > Share to Social |
| HTML | WEB | HTML > Web |
| HTML | VIRAL | HTML > Viral |
| HTML | WEB_MOBILE | HTML > Mobile Web |
| HTML | 2048 | HTML > Keitai Mail |
| HTML | ALL_EMAIL_STYLE_USAGE_MASK | Encompasses the following HTML options: Email Message, iPhone, Share to Social, Web, Viral, and Mobile Web. |
| HTML | ALL_WEB_STYLE_USAGE_MASK | Encompasses the following HTML options: Web, Share to Social, Mobile Web, and iPhone. |
| HTML | REPORT_SOCIAL_MASK | Encompasses the following HTML options: Share to Social, Viral. |
| HTML | REPORT_MSG_MASK | Encompasses the following HTML options: Email Message, Web, Mobile Web, and iPhone. |
| HTML | SUMMARY | Inserts this message content into the "Summary" field on the Campaign screen (see below for details). |
| HTML | NONE | Creates a "blank" Content Block with no format versions (Please note that the user interface doesn't allow you to create a Content Block with no format versions). |
| HTML | ALL | All possible usage masks. |
| TEXT | EMAIL | Plain Text > Email Message |
| TEXT | WEB_SOCIAL | Plain Text > Social Text |
| TEXT | WEB | Plain Text > Web |
| TEXT | 1024 | Plain Text > Push Notification |
| TEXT | VIRAL | Plain Text > Viral |

| type_id | usage_mask | User Interface Name |
|---|---|---|
| TEXT | 2048 | Plain Text > Keitai Mail |
| TEXT | ALL_EMAIL_STYLE_USAGE_MASK | Encompasses the following Plain Text options: Email Message, Web, Viral, and Social Text. |
| TEXT | ALL_WEB_STYLE_USAGE_MASK | Encompasses the following Plain Text options: Web, Social Text. |
| TEXT | REPORT_SOCIAL_MASK | Encompasses the following Plain Text options: Viral, Social Text. |
| TEXT | SMS | Mobile Text |
| TEXT | HTTP_REQUEST | Data > HTTP |
| TEXT | DATA_FILE | Data > Flat File |
| TEXT | NONE | Creates a "blank" Content Block with no format versions (Please note that the user interface doesn't allow you to create a Content Block with no format versions). |
| TEXT | ALL | All possible usage masks. |
| XML | WEB | XML |

As described above, the "SUMMARY" **usage_mask** value is used to populated the "Summary" field on the Campaign details screen.




The "Summary" field is relevant only if you're utilizing a Facebook Like Button within your message content. This Summary consists of a short message that appears on the interstitial webpage after the recipient clicks the Like button. If you provide the Summary content via the CONTENT BLOCK endpoint, you must then make this Content Block the

primary content source for the Campaign. When you then save the Campaign, the "Summary" field will be populated with the content provided via the request message.

**body**

This string parameter is optional.

The **body** parameter is used to provide the actual message content for this format version, such as the HTML code or plain text.

Example:

```
"body": "<html> <body> Hello, {(first_name)}!<br/><br/> Thank you for
your order!</body> </html>"
```

### contModelProps

This object is required in the message, but it's not currently used for anything. Simply include the empty object in the message.

Example:

```
"contModelProps": []
```

### contParts

This object is required in the message, but it's not currently used for anything. Simply include the empty object in the message.

Example:

```
"contParts": []
```

### obj

This object contains the name and location of the new Content Block.

Example:

```
"obj":
  {
  "display_name": "Sample Content Block",
  "parent_obj_id": 37249
  }
```

The parameters in this object are described below in more detail.

### display_name

This string parameter is required.

This parameter contains the name of the Content Block. This name must be unique within the selected folder location.

Example:

```
"display_name": "Sample Content Block"
```

### parent_obj_id

This integer parameter is required.

The **parent_obj_id** parameter represents the **Folder ID** of the folder where you want to save the new Content Block.

Example:

```
"parent_obj_id": 37249
```

# 3 Edit a Content Block

## Overview

This section describes the options for viewing, modifying, and deleting Content Blocks via the CONTENT BLOCK endpoint.

## Retrieve a Content Block

The GET method is used to retrieve all of the information about a specified Content Block.

When submitting a GET request to the CONTENT BLOCK endpoint, the request message must include the Content Block's **Object Reference ID** as a query type parameter within the URL.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?id=3456
```

## Delete a Content Block

The DELETE method is used to delete a specified Content Block.

When submitting a DELETE request to the CONTENT BLOCK endpoint, the request message must include the Content Block's **Object Reference ID** as a query type parameter within the URL.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?id=3456
```

# Edit a Content Block

The PUT method allows you to submit modifications to an existing Content Block. Using this method, you can change the Content Block name, modify the message content, and add or remove format versions. The parameters for the PUT method are the same as described in the **Create a Content Block** section.

To remove a format version from an existing Content Block, simply omit it from the request message; any existing format versions that aren't referenced in the PUT request message will be removed from the Content Block.

When submitting a PUT request to the CONTENT BLOCK endpoint, the request message must include the Content Block's **Object Reference ID** as a query type parameter within the URL:

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?id=3456
```

# Run Content Editor Tools

In addition to using a POST method to create a new Content Block (see **Create a Content Block** for details), you can also use the POST method to run the platform's built-in content editor tools.

### Convert HTML to Text

Messaging features an integrated HTML-to-Text converter that strips out the HTML tags from the HTML format version of a Content Block, in order to make a Plain Text format version. Using a POST message, you can run the HTML-to-Text convertor on a specified format version.

The system will automatically create a new Plain Text format version, and populate it with the results of the HTML-to-Text conversion. By default, this new Plain Text format version will have a **usage_mask** value of "ALL_EMAIL_STYLE_USAGE_MASK," which encompasses the Email Message, Web, Viral, and Social Text options.

The POST request message must include the following parameters. All of these parameters are query type parameters that must be sent as part of the URL.

#### contId

This integer parameter is required.

The **contId** parameter represents the **Object Reference ID** of the Content Block.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?contId=57940&contBody
UsageMask=WEB
```

#### contBodyUsageMask

This string parameter is required.

The **contBodyUsageMask** represents the existing HTML format version, in the specified Content Block, that you want to convert to plain text.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?contId=57940&contBody
UsageMask=WEB
```

### Convert CSS Styles to Inline Tags

Some consumer email applications don't support the use of a Cascading Style Sheet (CSS) when displaying HTML content. Messaging features an integrated tool that will convert CSS styles into "inline" HTML tags.

The POST request message must include the following parameters. All of these parameters are query type parameters that should be sent as part of the URL.

#### contId

This integer parameter is required.

The **contId** parameter represents the **Object Reference ID** of the Content Block.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?contId=57940&contBody
UsageMask=WEB&cssToInlineStyle=true
```

### contBodyUsageMask

This string parameter is required.

The **contBodyUsageMask** represents the existing HTML format version, in the specified Content Block, that you want to convert from CSS to inline styles.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?contId=57940&contBodyUsageMask=WEB&cssToInlineStyle=true
```

### cssToInlineStyle

This Boolean parameter is required.

The **cssToInlineStyle** parameter must be set to "true" to run the CSS conversion tool on the specified format version.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock?contId=57940&contBodyUsageMask=WEB&cssToInlineStyle=true
```

# 4 Pick Up Changes

## Overview

The Pick Up Changes feature allows you to make modifications to a Content Block used within a launched Campaign, while messages are still in the process of being deployed to the Campaign recipients.

When running Pick Up Changes, the platform suspends the current version of the Campaign, copies it into a new version, applies the updated content, then launches that new version containing the updated content. All messages not yet deployed will be updated to include the changes. Please note that any messages that have already been sent can't be retrieved.

Using the CONTENT BLOCK endpoint, you can execute the Pick Up Changes process on all Campaigns that use a specified Content Block.

If you need to make changes to a Content Block in order to update the content in a launched Campaign, you typically want to check first to see what the potential impact is of this change, by identifying the Campaigns that are using the Content Block. The CONTENT BLOCK endpoint supports two different ways of retrieving a list of impacted Campaigns:

- Retrieve a list of all Campaigns (in any status) that use a specified Content Block.
- Retrieve a list of all Campaigns that utilize a specified Content Block, and that have launched but aren't yet finished deploying; this list represents the Campaigns that would be impacted by the Pick Up Changes process at this point in time.

## Retrieve List of All Campaigns

The GET method is used to retrieve a list of all Campaigns, in any status, that use a specified Content Block. The Content Block can either be the primary content source for the Campaign, or inserted into the message content.

When submitting a GET request to the CONTENT BLOCK endpoint, the request message must include the Content Block's **Object Reference ID** as a path type parameter within the URL.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock/57940/RelatedCampaigns
```

## Retrieve List of Launched Campaigns

The GET method is used to retrieve a list of all Campaigns that use a specified Content Block, and that have launched, but haven't yet finished deploying. The Content Block can either be the primary content source for the Campaign, or inserted into the message content.

The Campaigns in the response message represent the Campaigns that would be impacted by the Pick Up Changes process, if you were to execute it right now.

When submitting a GET request to the CONTENT BLOCK endpoint, the request message must include the Content Block's **Object Reference ID** as a path type parameter within the URL.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock/57940/PickUpChanges
```

## Run Pick Up Changes

The PUT method is used to execute the Pick Up Changes process for all in-flight Campaigns that use a specified Content Block. The Pick Up Changes process will impact all Campaigns associated with this Content Block, and that have launched, but currently aren't yet finished deploying (the endpoint described above can be used to retrieve a list of all these impacted Campaigns).

Please note that this endpoint will automatically run Pick Up Changes on all impacted Campaigns. For example, let's say you've used a Content Block in five Campaigns that have launched, but currently haven't finished deploying. You've made changes to the Content Block, that you now want to push out to all remaining unsent messages. If you use this API endpoint, the system will automatically run Pick Up Changes on all five Campaigns. You can't, for example, tell the platform to run Pick Up Changes on only one Campaign, and not the other four. If you need that kind of granular control over which Campaigns to change, you'll need to log into the application, and manually run Pick Up Changes through the user interface.

When submitting a PUT request to the CONTENT BLOCK endpoint, the request message must include the Content Block's **Object Reference ID** as a path type parameter within the URL.

For example:

```
https://api.eccmp.com/services2/api/ContentBlock/57940/PickUpChanges
```

# 5 Response

This section describes the possible response messages sent back from the CONTENT BLOCK endpoint.

## Success

A successful response to a POST message to create a new Content Block will generate a response code of "200," followed by the details of the new Content Block contained within the body of the response message.

A successful response to a GET message to retrieve a Content Block will generate a response code of "200," followed by the details of the specified Content Block contained within the body of the response message.

A successful response to a PUT message to update a Content Block will generate a response code of "200," followed by the details of the modified Content Block contained within the body of the response message.

A successful response to a DELETE message will generate a response code of "204;" the body of the response message will be empty.

A successful response to a POST message to run either of the content editor tools on a Content Block will generate a response code of "200," followed by the details of the modified Content Block contained within the body of the response message.

A successful response to a GET message to retrieve a list of Campaigns associated with a Content Block will generate a response code of "200," followed by the Object Reference IDs of the relevant Campaigns.

A successful response to a POST message to run the Pick Up Changes process will generate a response code of "200," followed by the Object Reference IDs of the impacted Campaigns.

## Errors

If Messaging encounters a problem with a CONTENT BLOCK request message, the platform will send an "error" message with details of the problem. Below is a list of error codes and their descriptions.

| Response Code | Error message | Description |
|---|---|---|
| 400 | "Each content body must have at least one Use" | Required parameter **usage_mask** is missing or contains an invalid value. |
| 400 | "An Obj with this name already exists" | A Content Block with this same name already exists within the designated folder; the **display_name** value must be unique within a folder. |
| 500 | "[GetRelatedCampaigns] Obj could not be found" | For any of the Pick Up Changes related endpoints, the specified Object Reference ID does not exist. |

# 6 Sample Messages

This section contains a sample request message for the CONTENT BLOCK endpoint.

## Sample Request

This sample POST request message creates a new Content Block with two format versions.

*JSON Payload*

```
{
  "cust_id": 394,
  "entity_id": 100,
  "type_id": "PARAGRAPH",
  "contBodies":
    [
      {
      "type_id": "HTML",
      "usage_mask": "ALL_EMAIL_STYLE_USAGE_MASK",
      "body": "<html> <body> HTML Message content goes here. </body>
</html>"
      },
      {
      "type_id": "TEXT",
      "usage_mask": "EMAIL",
      "body": "Plain Text message content goes here."
      }
    ],
  "contModelProps": [],
  "contParts": [],
  "obj":
    {
    "display_name": "API Content Block",
    "parent_obj_id": 37249
    }
}
```

# 7 Appendix A -- Identifiers

Messaging uses several different types of IDs when referencing assets, such as tables, fields, folders, Filters, and so forth. This appendix describes these different types of IDs, and provides steps on how to look up the value of an ID.

## Entity ID

The Entity ID is a unique, system-generated identifier for every table in your database. This value is not displayed within the application user interface anywhere, so to get the Entity ID for a table, you must retrieve it by means of the TABLE API endpoint.

To retrieve the Entity ID for a table:

1. Submit a request to the TABLE API endpoint. The simplest method is to use the version of the TABLE endpoint that allows you to retrieve information based on the table's name. For example:

```
https://api.eccmp.com/services2/api/Table?tableName=recipient
```

2. Within the API response message, the system lists every field in this table. As part of the field details, the response message provides the Entity ID for this table.

Sample Response:

```
{
  "viewId": 1002,
  "entityId": 100,
  "displayName": "create_date",
  "propId": 1030,
  "columnName": "create_date"
}
```

For more details on the TABLE endpoint, please see the Messaging Online Help system or the *Messaging -- Table API Technical Guide*.

## Folder ID

The Folder ID is a unique, system-generated identifier for each folder and sub-folder in your system. This value is not displayed within the application user interface anywhere, so to get your Folder ID, you must retrieve it by means of the SEARCH API endpoint.

1. Submit a GET request to the SEARCH endpoint. The easiest method is to use the version that lets you search by object type -- use a type value of "Folder." For example:

```
https://api.eccmp.com/services2/api/Object?type=Folder
```

2. The response message provides a list of all the folders in your system. Find the desired folder in the response message.
3. As part of the API response message, the system provides the Folder ID, which is referred to as the "obj_id."

> **Note**
>
> If this Folder is a sub-folder, the "parent_obj_id" is the Folder ID of the parent folder.

Sample Response:

```
{
  "obj_id": 37465,
  "display_name": "Content Block Folder",
  "type_id": "Folder",
  "ref_id": 37465,
  "parent_obj_id": 22817,
  "eligibility_status_id": "READY"
}
```

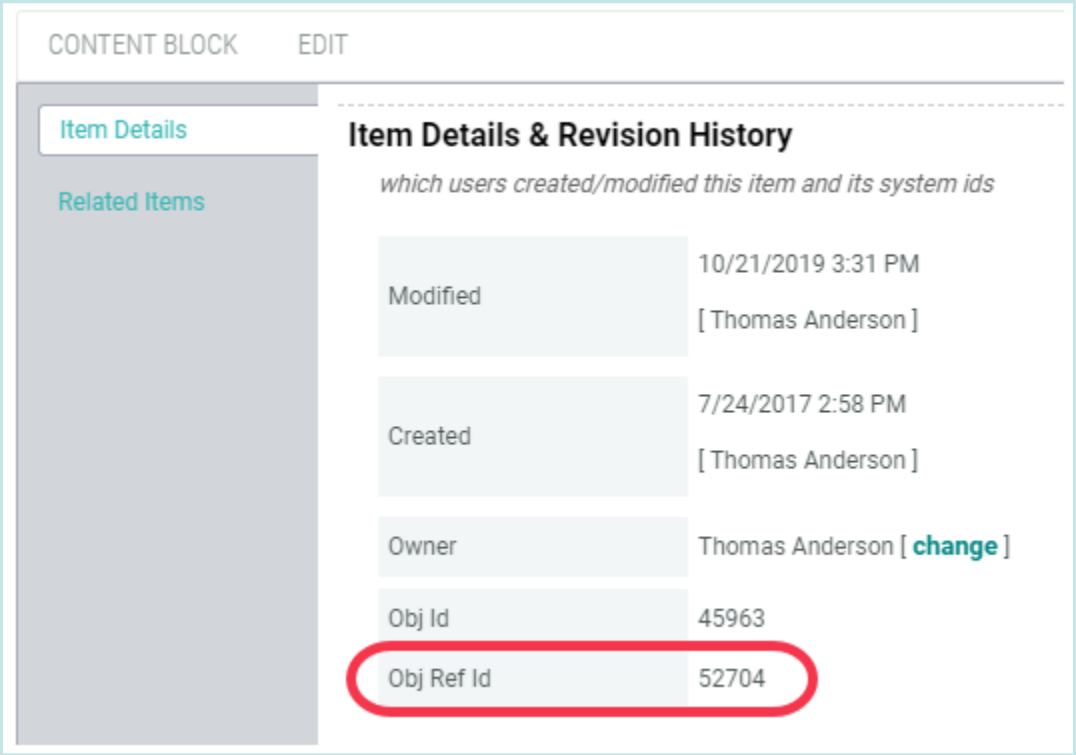
## Object Reference ID

The Object Reference ID is a system-generated identifier for every item and asset in your account.

The value for this identifier for a Content Block can be found within the Messaging application, or by using the SEARCH endpoint, which will return the Object Reference ID in the response message.

To find the Object Reference ID within the application:

1. From the System Tray, select *Assets* > *Management* > *Content Blocks*. The system displays a list of all the Content Blocks in your account.
2. Select the desired Content Block. The Content Block Details screen is displayed.
3. In the Tool Ribbon, click the " Content Block" tab. The "Item Details" screen is displayed. The Object Reference ID is listed on this screen.




Optionally, you can use the SEARCH endpoint, and search for the desired asset:

1. Submit a GET request to the SEARCH API endpoint. The simplest method is to use the versions of the SEARCH endpoint that allow you to retrieve information based on either the asset's name or its type. For example, to retrieve information about all of your Content Blocks:

```
https://api.eccmp.com/services2/api/Object?type=ContentBlock
```

2. The response message provides a list of all the assets in your system that match the search criteria. Find the desired asset in the response message.
3. As part of the API response message, the system provides the Object Reference ID, which is referred to as the **ref_id**. For example:

```
{
  "obj_id": 44737,
  "display_name": "Gold Member Content",
  "type_id": "ContentBlock",
  "ref_id": 40329,
  "parent_obj_id": 43269,
  "eligibility_status_id": "READY"
}
```