UNIVERSITY OF CAMBRIDGE

Zendesk macros

A crash course for OSC staff

André Sartori

Office of Scholarly Communication, University of Cambridge, UK

Contents

By the end of this course, I hope you will be able to:

- Create a new Zendesk macro
- Edit a Zendesk macro
- Use Liquid Markdown to output ticket variables
- Use Liquid Markdown to build logic into your macros
- Format the output of your macros using Zendesk markup

Symbols and typographic conventions

The following symbols and typographic conventions are used in these slides:

- Emphasis: This is an important concept or keyword
- External links: `https://camacuk.zendesk.com` or Zendesk
- User input: `This is something you should type.`
- Macro output:

  Your macro printed this.

- Liquid output tags: `{{ ticket.requester }}`
- Liquid logic tags: `{% if this is true %}`
- Exercises: ✍ in the top right corner of the slide
- Exercise solutions: 👍 in the top right corner of the slide
- Tips: 💡 denotes a tip

Let's **create** a Zendesk macro!

- Log into Zendesk: `https://camacuk.zendesk.com/agent`
- Click the Admin icon () in the sidebar, then select Macros
- Click the Add macro button
- Enter the macro name: `Salute world`
- Click the Available for drop-down menu and select Me only
- Click Add action and choose the Comment/description option in the drop-down menu
- Enter the text of the macro: `Hello world!`
- Click Create to save your brand new macro

Let's **test** your new macro!

- Navigate to `https://camacuk.zendesk.com/agent/tickets/94292`
- Click the ⚡ Apply macro drop-down menu in the bottom bar and type the words "Salute world"
- Hit the Enter/Return key on your keyboard
- Apply the macro again. What happens?

Expected output

Zendesk adds the string "Hello world!" to the ticket body every time you activate your macro using the steps above.

Let's **edit** that macro!

- Click the Admin icon () in the sidebar, then select Macros
- Change the value of the filter All shared macros to Personal macros
- Click the Salute world macro
- Change the text produced by the macro to:

```
Dear world,
Please upload your paper via your Elements profile.
Yours,
Cambridge
```

Editing a macro you have just used

Use your macro on a test ticket

- Go to https://camacuk.zendesk.com/agent/tickets/94292
- Apply your Salute world macro
- Change your reply to Internal note and click Submit as On-hold

Quickly access the macro editing form via ticket Events

- Change the ticket view from Conversations to Events
- Look for event: **Macro applied** Salute world
- Click the link on that event. It will take you to this macro's edit form

Zendesk macros can do a lot more than that!

Actions

Navigate to the macro edit form again and check out the list of available actions. Commonly used options include:

- Set subject: Sets or changes the subject line of the ticket
- Status: Sets or changes the status (Open, Pending, etc) of the ticket
- Assignee: Sets or changes the ticket assignee (e.g. to current user)
- Comment mode: e-mail to requester (Public) or internal note (Private)
- Custom ticket fields: Sets or changes the value of ticket fields

You can add as many actions to a macro as necessary. All will be performed, each in the order they were added.

Zendesk macros can do a lot more than that!

Liquid markup

Macro e-mails support dynamic content via Liquid, a templating language consisting of two types of markup:

- Output markup is surrounded by

  ```
  {{ matched pairs of curly brackets }}
  ```

  When the macro is run, this gets replaced by the value of the expression between the braces. For instance, `{{ticket.id}}` gets replaced by the ticket number and `{{ticket.requester.email}}` by the e-mail address of the ticket requester.

- Tag markup is surrounded by

  ```
  {% matched pairs of curly brackets and percent signs %}
  ```

  Tags cannot resolve to text. They are used for the logic of your macro.

Let’s make your macro more general!

Replacing static text by variables

Edit your macro so that:

1. The e-mail is addressed to the ticket requester rather than to “world”
2. It is signed by whoever has activated the macro, rather than by “Cambridge”

Tip

Click View available placeholders just below the box containing the macro text in the edit form to see a list of supported output markup statements

Here is a possible solution

```
Dear {{ ticket.requester.name }},

Please upload your paper via your Elements profile.

Yours,

{{ current_user.first_name }}
```

Outputing the value of custom ticket fields

Definition

Custom fields are all ticket fields that were created by our team rather than shipped with Zendesk (e.g. #Journal title)

Syntax

Use the following syntax to output the value of a custom ticket field:

```
{{ ticket.ticket_field_ID }}
```

Where ID is a unique numeric identifier of the field of interest.

List of ticket field IDs

You can find out the ID of all ticket fields at
`https://camacuk.zendesk.com/agent/admin/ticket_fields`

Edit your macro to output the publication title

After using your macro for one week, you received three replies asking you what paper you are talking about. You decided it would be a good idea to output the title of the paper near the top of your message:

> *Dear Dr Sartori,*
>
> *Article title: Zendesk macros are useful*
>
> *Please upload your paper via your Elements profile.*
>
> *Yours,*
>
> *André*

Edit your macro to output the value of ticket field #Manuscript title, matching the highlighted line of the example above.

Here is the solution

```
Dear {{ ticket.requester.name }},

Article title:  {{ ticket.ticket_field_24069473 }}

Please upload your paper via your Elements profile.

Yours,

{{ current_user.first_name }}
```

Building logic into your macros

Conditional statements

You can also instruct your macro to only produce output if a certain condition is met, such as a particular ticket checkbox being ticked or a dropdown menu having a certain value.

Syntax

```
{% if CONDITION %}...{% endif %}
{% if CONDITION %}...{% elsif CONDITION %}...{% else %}...{% endif %}
```

Where `CONDITION` is usually a comparison between the current value of a ticket field and a constant using the operators:

- `==` for equality
- `!=` for inequality
- `<` for less than
- `<=` for less than or equality
- `>` for greater than
- `>=` for greater than or equality

Building logic into your macros

Conditional statements—Drop-down fields

When `CONDITION` involves a drop-down field, comparisons are made to tags associated with each value. The word `blank` is used to denote that the drop-down field has not been populated (null value).

Here is an example using field “Is there an APC payment?”:

```
{% if ticket.ticket_field_24071633 == 'springer_compact' %}
    Your paper is covered by Springer Compact.
{% elsif ticket.ticket_field_24071633 == 'payment_yes' %}
    Please request an invoice for OA charges using the
billing address below.
{% elsif ticket.ticket_field_24071633 == blank %}
{% else %}
    There is no need to select a paid-for Open Access
option.
{% endif %}
```

Building logic into your macros

Conditional statements—Checkbox fields

When `CONDITION` involves a checkbox field:

- ‘0’ denotes the checkbox is not ticked
- ‘1’ denotes the checkbox is ticked

Here is an example using field “RCUK policy”:

```
{% if ticket.ticket_field_24071783 == '1' %}
    Your paper is covered by the RCUK Open Access policy.
{% endif %}
```

Building logic into your macros

Conditional statements—Date fields

When `CONDITION` involves a date field, the date you will compare the value of the field to should be written in the format 'YYYY-MM-DD'

Here is an example using field "#Acceptance date":

```
{% if ticket.ticket_field_24114776 >= '2016-04-01' %}
    Your paper is covered by the REF Open Access policy.
{% elsif ticket.ticket_field_24114776 == blank %}
    Please send us the acceptance date of your paper.
{% endif %}
```

Building logic into your macros

Combining conditional statements

It is possible to write a `CONDITION` that involves more than one test by using the operators:

- `and` when you want to produce output only when all tests succeed
- `or` if you want to produce output when at least one test succeed

Building logic into your macros

Combining conditional statements—Examples

Here is an example using the operator or to test for two different values of field “Is there an APC payment?”:

```
{% if ticket.ticket_field_24071633 == 'springer_compact' or
ticket.ticket_field_24071633 == 'payment_yes' %}
    We will be able to cover Open Access charges for your
paper.
{% endif %}
```

Here is an example combining fields “#Publication date” and “#Online Publication Date” with the operator and:

```
{% if ticket.ticket_field_24069583 == blank and
ticket.ticket_field_47509168 == blank %}
    Your paper is under an indefinite embargo as it has not
been published yet.
{% endif %}
```

Nesting conditional statements

It is not possible to combine tests using both operators (and, or) in the same CONDITION, but you work around that limitation by nesting conditional statements. For example:

```
{% if ticket.ticket_field_24071633 == 'springer_compact' or
ticket.ticket_field_24071633 == 'payment_yes' %}
   {% if ticket.ticket_field_24069583 == blank and
ticket.ticket_field_47509168 == blank %}
      We will be able to cover Open Access charges for your
paper, which has not been published yet.
   {% endif %}
{% endif %}
```

Add conditional statements to your macro

Edit your macro so that it outputs:

- “Please upload the accepted version of your paper...” if field Wrong version is ticked
- an extra sentence if field #Online Publication Date has a value: “Although your paper is already published, please try to resist the urge of sending us the published version.”
- “There is no need to upload your conference abstract to us” if field publication type has value conference abstract

List of ticket field IDs and tags

Remember that you can find out the ID and tags of all ticket fields at `https://camacuk.zendesk.com/agent/admin/ticket_fields`

Here is a possible solution

```
Dear {{ ticket.requester.name }},

Article title:  {{ ticket.ticket_field_24069473 }}

{% if ticket.ticket_field_80501387 == 'conference_abstract'
%}There is no need to upload your conference abstract to us.
{% else %} Please upload {% if ticket.ticket_field_24069523
== '1' %}the accepted version of{% endif %}your paper via
your Elements profile.  {% if ticket.ticket_field_47509168 !=
blank %}Although your paper is already published, please try
to resist the urge of sending us the published version.{%
endif %}{% endif %}

Yours,

{{ current_user.first_name }}
```

Making your macro easier to understand

Or, how to invest a little of your time now to avoid much pain later

Let's focus for a moment on the structure of the main paragraph of your macro:

```
{% if ticket.ticket_field_80501387 ==
'conference_abstract' %}...{% else %}...{% if
ticket.ticket_field_24069523 == '1' %}...{% endif
%}...{% if ticket.ticket_field_47509168 != blank
%}...{% endif %}{% endif %}
```

It is hard to visualise the logical structure even when we replace all text with . . .

Making your macro easier to understand

Or, how to invest a little of your time now to avoid much pain later

This is much easier to understand:

```
{% if ticket.ticket_field_80501387 == 'conference_abstract' %}
    ...
{% else %}
    ...
    {% if ticket.ticket_field_24069523 == '1' %}
        ...
    {% endif %}
    ...
    {% if ticket.ticket_field_47509168 != blank %}
        ...
    {% endif %}
{% endif %}
```

But...

Making your macro easier to understand

Or, how to invest a little of your time now to avoid much pain later

White space and line breaks are interpreted as such:

```
...

    Please upload

        the accepted version of

    your paper via your Elements profile.

...
```

It might be hard to remember what a condition means:

```
{% if ticket.ticket_field_24069523 == '1' %}
{% if ticket.ticket_field_47509168 != blank %}
```

Commenting your macro code

Comments are completely ignored by Zendesk, so they can be used to annotate and/or indent parts of your macro.

Syntax

```
{% comment %}...{% endcomment %}
```

Example

```
{% if ticket.ticket_field_24069523 == '1' %}{% comment %}
   BEGIN: IF 'WRONG VERSION' IS TICKED
   {% endcomment %}the accepted version of{% comment %}
   END: IF 'WRONG VERSION' IS TICKED
{% endcomment %}{% endif %}
```

Your turn

Add comments to your macro to make more explicit what the following if statements are testing for:

```
{% if ticket.ticket_field_24069523 == '1' %}
{% if ticket.ticket_field_47509168 != blank %}
```

Here is a possible solution

```
...
{% if ticket.ticket_field_24069523 == '1' %}{% comment %}
    BEGIN: IF "WRONG VERSION" IS TICKED
    {% endcomment %}the accepted version of{% endif %}{% comment %}
    END: IF "WRONG VERSION" IS TICKED
{% endcomment %}your paper via your Elements profile.  {% if
ticket.ticket_field_47509168 != blank %}{% comment %}
    BEGIN: IF "#ONLINE PUBLICATION DATE" IS NOT NULL
    {% endcomment %}Although your paper is already published, please
try to resit the urge of sending us the published versio{% endif
%}{% comment %}
    END: IF "#ONLINE PUBLICATION DATE" IS NOT NULL
{% endcomment %}
...
```

Syntax highlighting

Here, I have used colours to faciliate the distinction of Liquid statements from regular text. You can obtain a similar effect in Notepad++ by:

- Download this file: `https://github.com/osc-cam/zendesk/blob/development/macros/notepad%2B%2B_liquid_markup.xml`
- Open Notepad++ and:
  - Click Language in the top menu and select option Define your language...
  - Click the button Import... near the top of the "User Defined Language" window
  - Choose the file you downloaded above and click the Open button. A popup reading "Import successful" should appear
  - Click the button Save As... near the top of the "User Defined Language" window and enter a sensible name (Liquid; Zendesk; etc)

Now, the next time you click Language in the top menu of Notepad++, the option to select the language you just defined should appear at the end of the list. Selecting this language will allow you to view Liquid markup syntax in colour.

Formatting your macro's output with Markdown

You can use the following notation to format the output of your macro:

Formatting	User input	Macro output
Bold	`This is how you **bold** text.`	This is how you **bold** text.
Italics	`This is *italicized* text.`	This is *italicized* text.

Headings	`# Level one`
	`## Level two`
	`### Level three`
	`...and so on up to level six`
		Level one
		Level two
		Level three

Links	`[Display text](http://www.sampleurl.com)`	Display text
Bulleted lists	`* Bullet one` `* Bullet two`	• Bullet one • Bullet two
Numbered lists	`1. Step one` `2. Step two`	1. Step one 2. Step two

Tip

Markdown works not only in macros, but also in all your replies. You can preview your formatting before sending an e-mail by clicking the Preview button of a Zendesk ticket.

Prettify your macro!

Use Markdown to prettify your macro in the following way:

- Output the string “Article title” in **bold**
- The string “Elements profile” should contain a clickable link to `https://elements.admin.cam.ac.uk`

Here is the solution

```
Dear {{ ticket.requester.name }},

**Article title**:  {{ ticket.ticket_field_24069473 }}

{% if ticket.ticket_field_80501387 == 'conference_abstract'
%}There is no need to upload your conference abstract to us.
{% else %} Please upload {% if ticket.ticket_field_24069523
== '1' %}the accepted version of{% endif %}your paper via
your [Elements profile](https://elements.admin.cam.ac.uk).
{% if ticket.ticket_field_47509168 != blank %}Although your
paper is already published, please try to resist the urge of
sending us the published version.{% endif %}{% endif %}

Yours,

{{ current_user.first_name }}
```

House rules and recommendations

To wrap up this crash course, let's go through some suggestions of good practice regarding our collection of macros:

- Share your macros by setting "Available for" to "Agents in group"
- Place your macros in a sensible folder by using this syntax to name your macro:
  `Folder :: Subfolder :: Macro name`
- Do not be afraid to edit shared macros. Remember:
  The only way you can avoid mistakes is by not doing anything!
- The macro Open access :: General reply is exceptional in complexity. To edit this macro you will need to use GitHub. See `https://help.github.com/en/desktop/getting-started-with-github-desktop` to get started.

Where to find more information

This crash course covered only the most frequently used features of Zendesk macros and Liquid. To learn more about the following topics, please refer to these links:

Liquid

https://github.com/Shopify/liquid/wiki/Liquid-for-Designers

How to format the output text, add external links, etc

https://support.zendesk.com/hc/en-us/articles/203691016-Formatting-text-with-Markdown