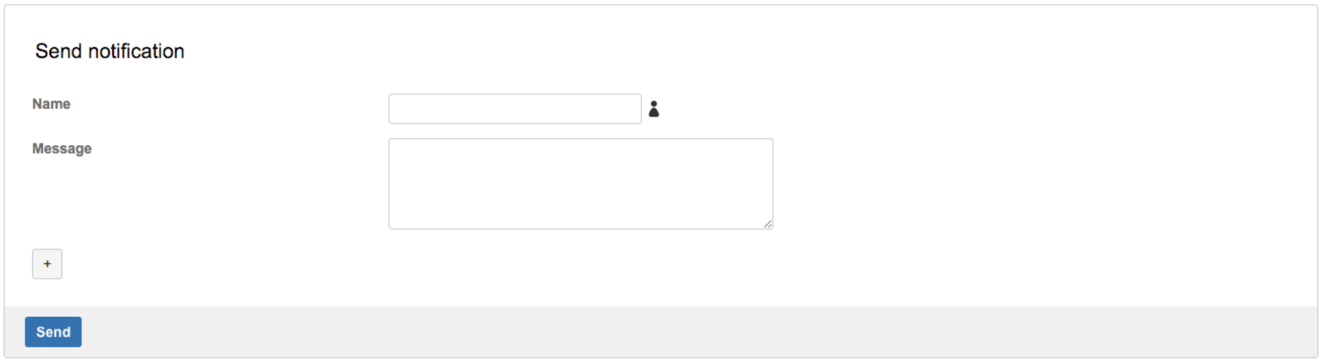
Create a form with ConfiForms which sends internal notification but does not keep/store the data

In this tutorial you will learn how to create a form with ConfiForms which will:

- send Confluence notification to a user
- will not keep the data in the form (will delete it right after the notification is sent)

If you are new to ConfiForms, please take a few minutes to read the Basic concepts Guide, or go through the video tutorial on that page. 🙂

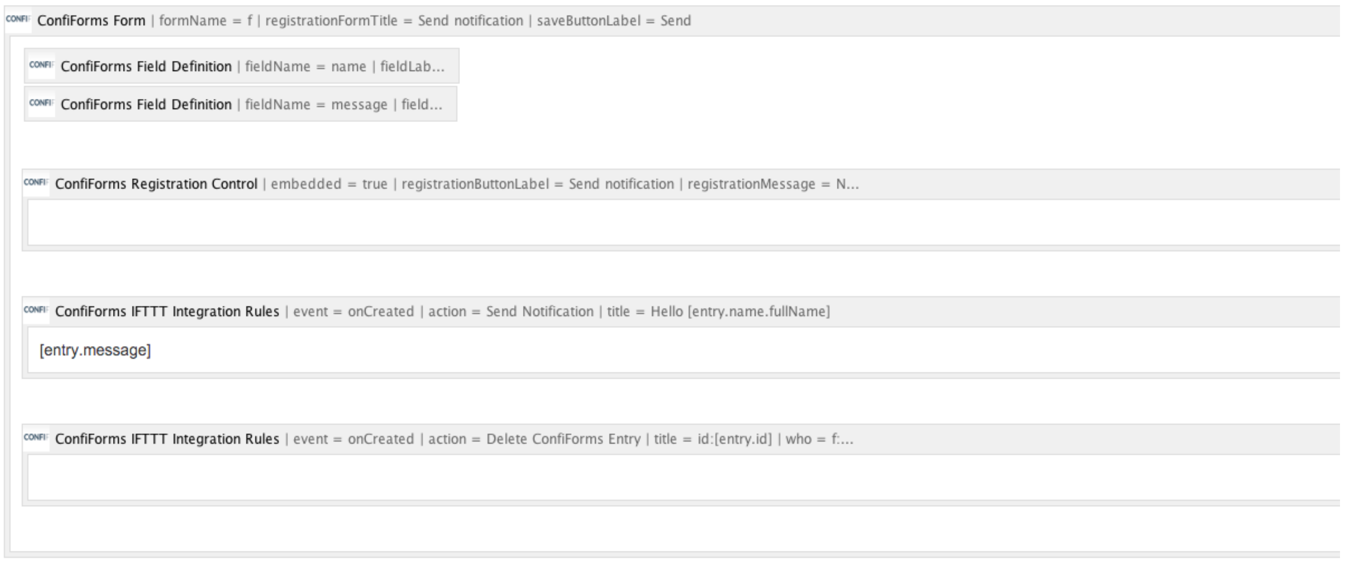
This is how the final form looks like



Complete storage format

```
<ac:structured-macro ac:macro-id="7090c376-21fb-4e40-9249-f6fe32aa62f3" ac:name="confiform" ac:schema-version="
1">
  <ac:parameter ac:name="formName">f</ac:parameter>
  <ac:parameter ac:name="saveButtonLabel">Send</ac:parameter>
  <ac:parameter ac:name="registrationFormTitle">Send notification</ac:parameter>
  <ac:rich-text-body>
    <p>
      <ac:structured-macro ac:macro-id="303ecca1-bb19-44c9-b188-990db8e3c711" ac:name="confiform-field-
definition"
                           ac:schema-version="1">
        <ac:parameter ac:name="fieldName">name</ac:parameter>
        <ac:parameter ac:name="fieldLabel">Name</ac:parameter>
        <ac:parameter ac:name="type">user</ac:parameter>
      </ac:structured-macro>
    </p>
    <ac:structured-macro ac:macro-id="c0a8c1cf-326d-4321-ab1f-8ef743bdc8d6" ac:name="confiform-field-definition"
                         ac:schema-version="1">
      <ac:parameter ac:name="fieldName">message</ac:parameter>
      <ac:parameter ac:name="fieldLabel">Message</ac:parameter>
      <ac:parameter ac:name="type">textarea</ac:parameter>
    </ac:structured-macro>
    <p>&nbsp;
      <ac:structured-macro ac:macro-id="052bc766-c50c-4396-a666-8f2436ef01bf" ac:name="confiform-entry-register"
                           ac:schema-version="1">
        <ac:parameter ac:name="registrationMessage">Notification has been sent</ac:parameter>
        <ac:parameter ac:name="registrationButtonLabel">Send notification</ac:parameter>
        <ac:parameter ac:name="embedded">true</ac:parameter>
        <ac:parameter ac:name="atlassian-macro-output-type">INLINE</ac:parameter>
        <ac:rich-text-body>
          <p>&nbsp;</p>
        </ac:rich-text-body>
      </ac:structured-macro>
    </p>
    <ac:structured-macro ac:macro-id="5f096006-a6fc-4243-8e71-8f3e09907c2e" ac:name="confiform-ifttt"
                         ac:schema-version="1">
      <ac:parameter ac:name="action">Send Notification</ac:parameter>
      <ac:parameter ac:name="event">onCreated</ac:parameter>
      <ac:parameter ac:name="title">Hello [entry.name.fullName]</ac:parameter>
      <ac:rich-text-body>
        <p>[entry.message]</p>
      </ac:rich-text-body>
    </ac:structured-macro>
    <ac:structured-macro ac:macro-id="cd72d761-5c5b-4fb3-bea5-9bb9c8496b67" ac:name="confiform-ifttt"
                         ac:schema-version="1">
      <ac:parameter ac:name="action">Delete ConfiForms Entry</ac:parameter>
      <ac:parameter ac:name="event">onCreated</ac:parameter>
      <ac:parameter ac:name="title">id:[entry.id]</ac:parameter>
      <ac:parameter ac:name="who">f:884778</ac:parameter>
      <ac:rich-text-body>
        <p>&nbsp;</p>
      </ac:rich-text-body>
    </ac:structured-macro>
  </ac:rich-text-body>
</ac:structured-macro>
```

And this is how it looks in the editor


ConfiForms Form | formName = f | registrationFormTitle = Send notification | saveButtonLabel = Send

ConfiForms Field Definition | fieldName = name | fieldLab...

ConfiForms Field Definition | fieldName = message | field...

ConfiForms Registration Control | embedded = true | registrationButtonLabel = Send notification | registrationMessage = N...

ConfiForms IFTTT Integration Rules | event = onCreated | action = Send Notification | title = Hello [entry.name.fullName]

[entry.message]

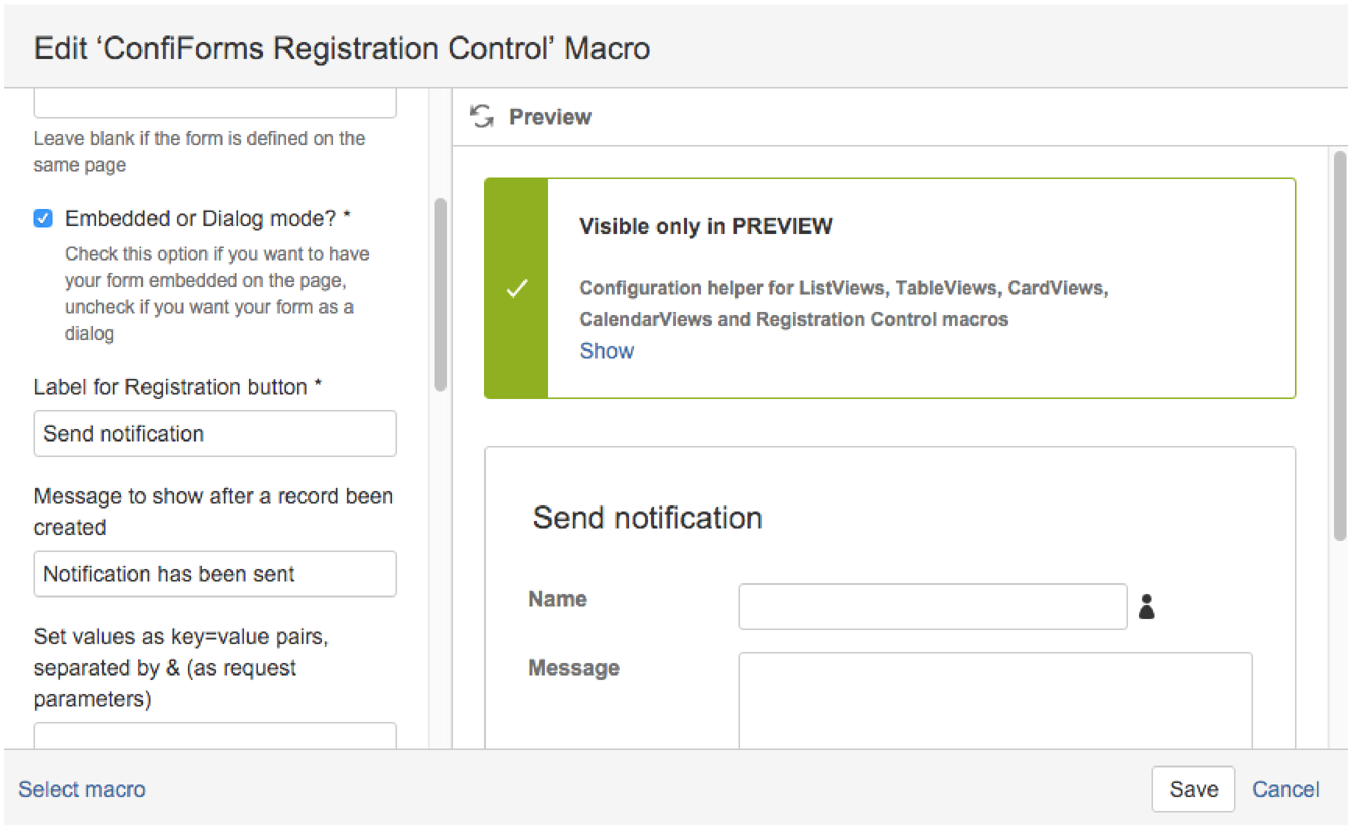
ConfiForms IFTTT Integration Rules | event = onCreated | action = Delete ConfiForms Entry | title = id:[entry.id] | who = f:...

Let's see what it consists of:

- As usual, everything is ConfiForms Form container, we have set custom title and custom label for main action button
- Then we have 2 fields (ConfiForms Field Definition macros): name and message. Name is of type "user" and Message is of type "textarea"
- Then we have a ConfiForms Registration Control, which tells us how the form should be shown (we have selected it to be embedded to the page) and we set custom messages and labels
- Then we have 2 ConfiForms IFTTT macros, or handlers as we usually call it. The ordering is important! The first one sends notification and the second one deletes the original ConfiForms record which was used initially to sent the notification

Below, you can see screenshots for Registration Control and screenshots for these 2 IFTTT handlers

Configuration for ConfiForms Registration Control


Edit 'ConfiForms Registration Control' Macro

Leave blank if the form is defined on the same page

☑ Embedded or Dialog mode? *

Check this option if you want to have your form embedded on the page, uncheck if you want your form as a dialog

Label for Registration button *

Send notification

Message to show after a record been created

Notification has been sent

Set values as key=value pairs, separated by & (as request parameters)

Preview

Visible only in PREVIEW

Configuration helper for ListViews, TableViews, CardViews, CalendarViews and Registration Control macros

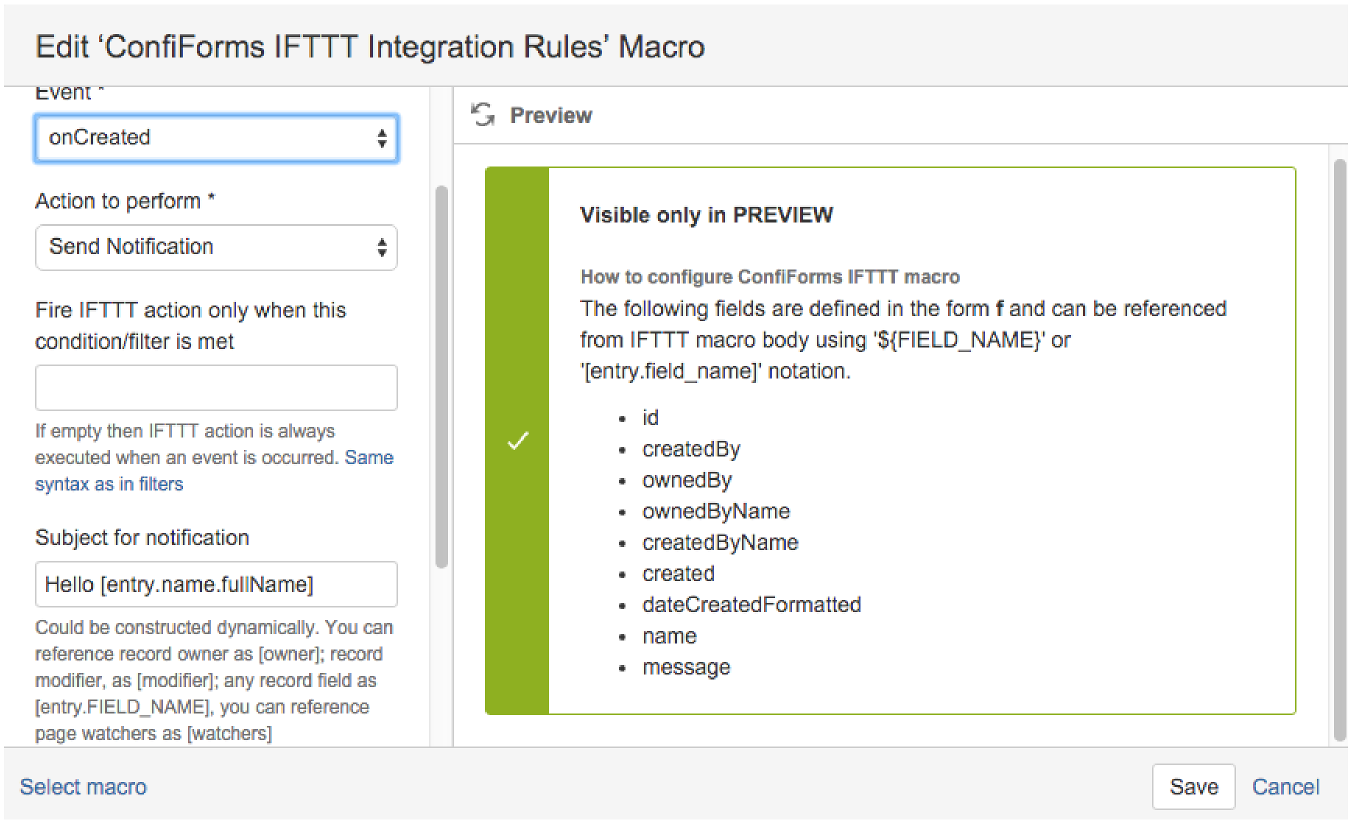
Show

Send notification

Name

Message

Configuration for ConfiForms IFTTT macro which sends notifications


Edit ‘ConfiForms IFTTT Integration Rules’ Macro

Event *

onCreated

Action to perform *

Send Notification

Fire IFTTT action only when this condition/filter is met

If empty then IFTTT action is always executed when an event is occurred. Same syntax as in filters

Subject for notification

Hello [entry.name.fullName]

Could be constructed dynamically. You can reference record owner as [owner]; record modifier, as [modifier]; any record field as [entry.FIELD_NAME], you can reference page watchers as [watchers]

Preview

Visible only in PREVIEW

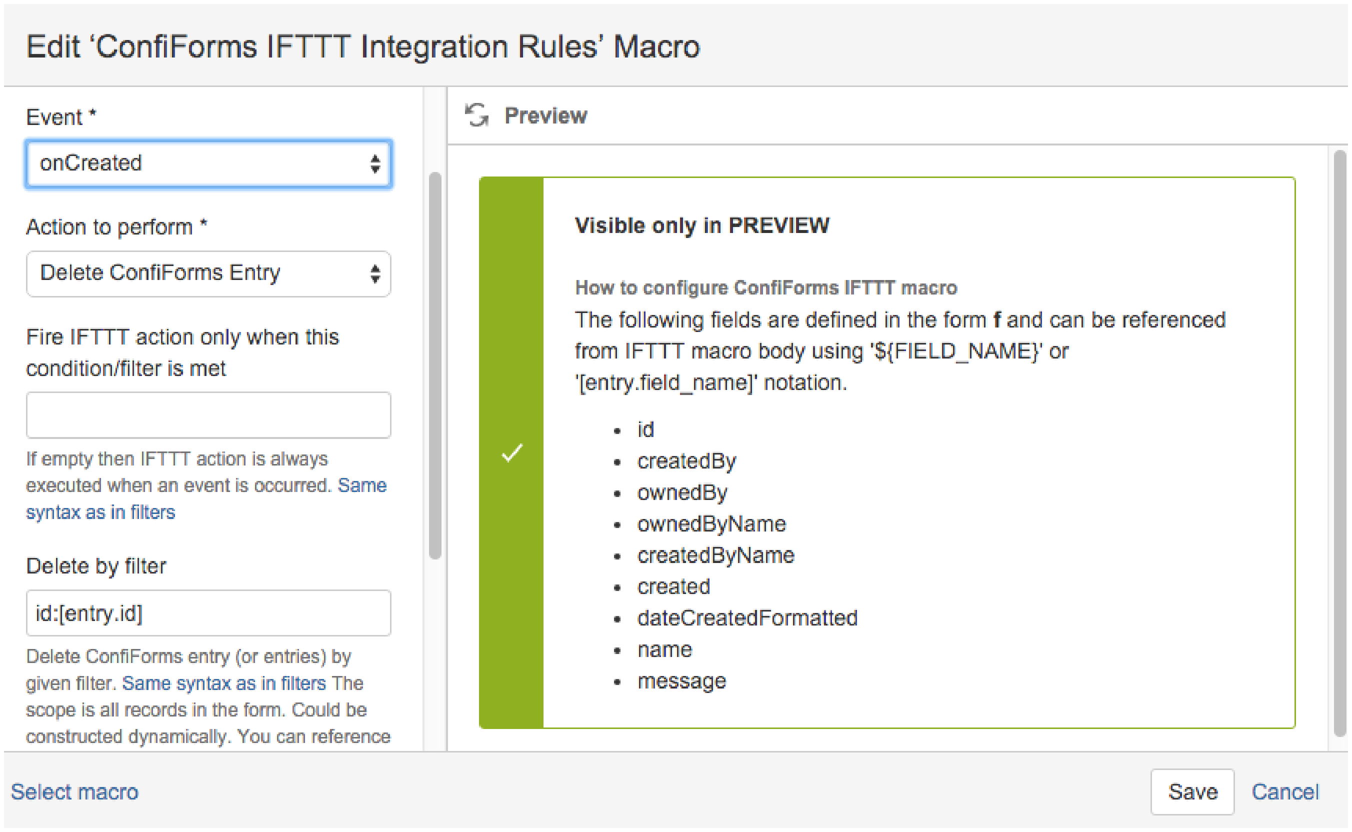
How to configure ConfiForms IFTTT macro

The following fields are defined in the form **f** and can be referenced from IFTTT macro body using '${FIELD_NAME}' or '[entry.field_name]' notation.

- id
- createdBy
- ownedBy
- ownedByName
- createdByName
- created
- dateCreatedFormatted
- name
- message

Here we set a subject to be dynamic, and as "user" object is a "complex" object we can query some additional properties (See documentation for more details on properties of "complex" objects: Documentation)

Second IFTTT deletes the record (we set a dynamic filter, which will filter only one record)


Edit 'ConfiForms IFTTT Integration Rules' Macro

Event *

onCreated

Action to perform *

Delete ConfiForms Entry

Fire IFTTT action only when this condition/filter is met

If empty then IFTTT action is always executed when an event is occurred. Same syntax as in filters

Delete by filter

id:[entry.id]

Delete ConfiForms entry (or entries) by given filter. Same syntax as in filters The scope is all records in the form. Could be constructed dynamically. You can reference

Preview

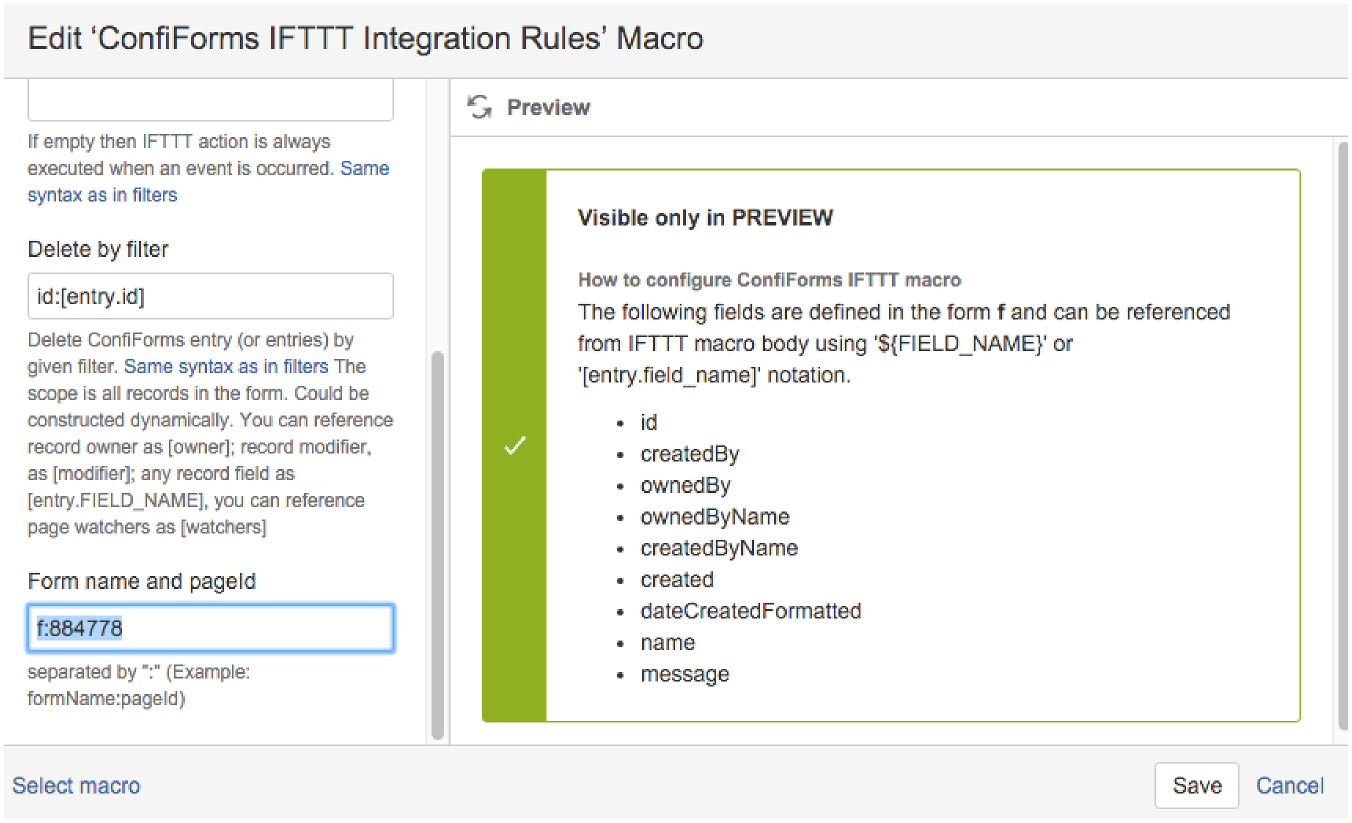
Visible only in PREVIEW

How to configure ConfiForms IFTTT macro

The following fields are defined in the form **f** and can be referenced from IFTTT macro body using '${FIELD_NAME}' or '[entry.field_name]' notation.

- id
- createdBy
- ownedBy
- ownedByName
- createdByName
- created
- dateCreatedFormatted
- name
- message

The important bit (shown below, as it did not fit into the previous screenshot) is the last parameter, called "Form name and pageId":


Edit 'ConfiForms IFTTT Integration Rules' Macro

If empty then IFTTT action is always executed when an event is occurred. Same syntax as in filters

Delete by filter

id:[entry.id]

Delete ConfiForms entry (or entries) by given filter. Same syntax as in filters The scope is all records in the form. Could be constructed dynamically. You can reference record owner as [owner]; record modifier, as [modifier]; any record field as [entry.FIELD_NAME], you can reference page watchers as [watchers]

Form name and pageId

f:884778

separated by ":" (Example: formName:pageId)

Preview

Visible only in PREVIEW

How to configure ConfiForms IFTTT macro

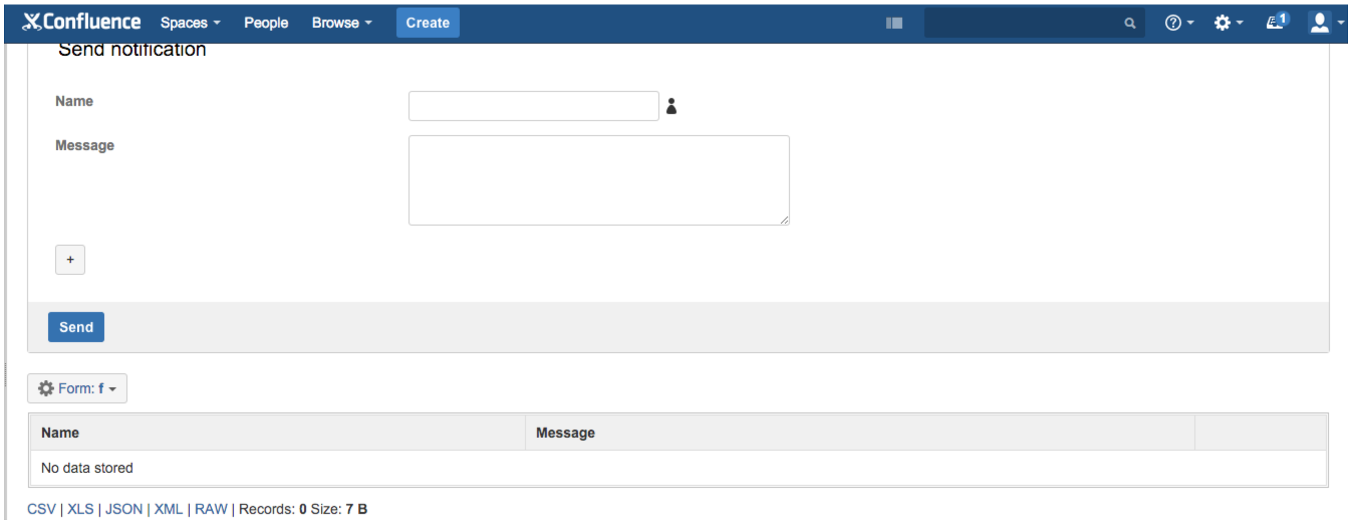
The following fields are defined in the form **f** and can be referenced from IFTTT macro body using '${FIELD_NAME}' or '[entry.field_name]' notation.

- id
- createdBy
- ownedBy
- ownedByName
- createdByName
- created
- dateCreatedFormatted
- name
- message

which set's the name of the form this filter (and action) should be applied on, as well as the location (pageId of the page where this form is configured)

In our case the form name is "f" and it is on the page with page Id = 884778

The end result looks like this (notification and actually empty table):



Send notification

Name	
Message	

Name	Message	
No data stored		

CSV | XLS | JSON | XML | RAW | Records: **0** Size: **7 B**

This will not work for "Anonymous forms"! The reason is: a record stored does not have an owner and therefore could not be deleted by anyone than a form administrator