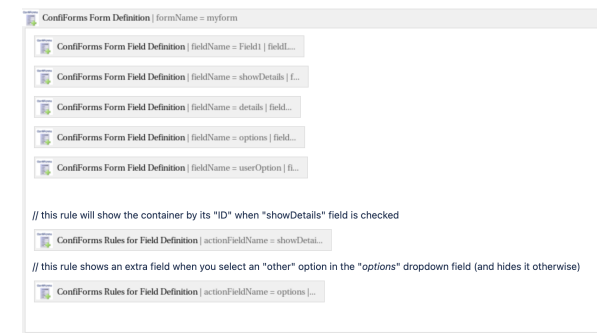


# How to show or hide blocks of fields in ConfiForms conditionally

Simple tutorial on how to use [ConfiForms Field Definition Rules#Showcontainer](#) and it's counterpart [ConfiForms Field Definition Rules#Hidecontainer](#) ConfiForms Rules for Field Definitions to show and hide visual blocks in your forms conditionally

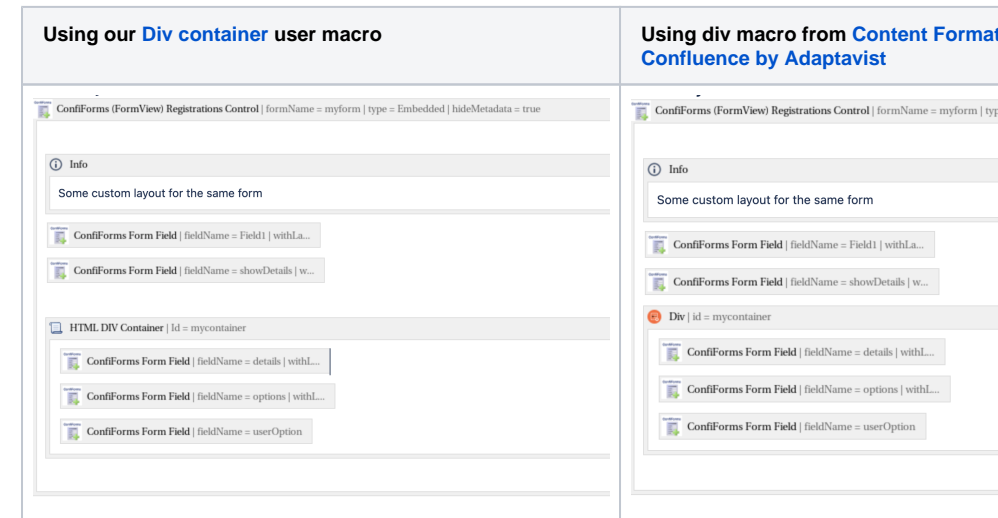
These rules are usually used with custom layouts, where you have your own layout defined for the form and want to show or hide blocks of fields conditionally

Consider a form that has few fields like this



## Custom layout for the same form

Implemented like



Please note that [Div container](#) user macro we have has certain limitations due to the way how user macros are rendered in Confluence. If you can use other macro that can be addressed via CSS then it is advisable to use that

## Configuration

As you can see, our form has 5 fields and 2 rules. Actually there are 4 rules, as each of the 2 rules has a "reverse rule" automatically created. That works for show/hide rules very well and you do not need to create an extra reverse rule yourself.

Our 5 fields are:

## Full configuration for the form in storage format

```
<ac:structured-macro ac:
macro-id="a356f4a8-bdfc-
4f79-bc45-56clabc82106" ac:
name="conform" ac:schema-
version="1">
    <ac:parameter ac:
name="formName">myform</ac:
parameter>
    <ac:rich-text-body>
    <p>
        <ac:structured-
macro ac:macro-id="
94923a9b-bc51-4272-a4f3-
e1711836ea84" ac:name="
conform-field-
definition" ac:schema-
version="1">
            <ac:
parameter ac:name="
fieldName">Field1</ac:
parameter>
            <ac:
parameter ac:name="
fieldLabel">Field1</ac:
parameter>
            <ac:
parameter ac:name="type"
>text</ac:parameter>
        </ac:
structured-macro>
    </p>
    <p>
        <ac:structured-
macro ac:macro-id="
faa26a33-b268-415f-bda2-
ca26354f8237" ac:name="
conform-field-
definition" ac:schema-
version="1">
            <ac:
parameter ac:name="
fieldName">showDetails</ac:
parameter>
            <ac:
parameter ac:name="
fieldLabel">Let me provide
some details</ac:parameter>
            <ac:
parameter ac:name="type"
>checkbox</ac:parameter>
        </ac:
structured-macro>
    </p>
    <p>
        <ac:structured-
macro ac:macro-id="
cfde56ca-1d74-40f5-a437-
105c7760d598" ac:name="
conform-field-
definition" ac:schema-
version="1">
```

- Field1 - simple text field (does not really needed here, but added to show a simplest field possible - no rules, no behaviour)
- showDetails - is a checkbox field and that manages the visibility of the other field named "details"
- details - textarea field that is shown only when the checkbox "showDetails" is checked
- options - is a dropdown field with 4 options, 4th option manages the visibility of the "userOption" field
- userOption - is a text field that is shown only when someone has selected an option with ID=4 in the "options" field

Now let's see the 2 rules that add the dynamics to our form:

```
// this rule will show the container by its "ID" when "showDetails" field is checked
<Confirms Rules for Field Definition | actionFieldName = showDetail...
// this rule shows an extra field when you select an "other" option in the "options" dropdown field (and hides it otherwise)
<Confirms Rules for Field Definition | actionFieldName = options |...
```

Rule 1	Rule 2
<p>First rule is "bound" to field "showDetails" and tracks for changes</p> <p>It has a condition to check if showDetails field has value "true"</p> <p>and we use CSS3 selector, by element id (more on CSS <a href="https://developer.mozilla.org/en-US/docs/Web/CSS">https://developer.mozilla.org/en-US/docs/Web/CSS</a>)</p> <div>#mycontainer</div> <p>As this is the name "mycontainer" we set to our DIV macro's ID parameter</p>	<p>See how it is configured here: <a href="#">How to show and hide fields in the form conditionally</a></p>
<div><h2>Edit 'ConfiForms Rules for Field</h2><hr/><h3>Rules for ConfiForms Field Definitions Documentation</h3><h4>Field name (or regular expression)</h4><div>showDetails</div><p>Name of the field you want to track for changes (leave blank when you configure validation rules, as these rules are applied on form submit). Parameter supports regular expressions to affect multiple fields with the same rule.Can be given as comma separated list of fields</p><h4>Condition</h4><div>showDetails:true</div><p>Action will be executed when the condition is met. <a href="#">Same syntax expected as in filters</a></p><p>Empty value means it matches everything.</p><p>The scope is <b>current entry/record</b> (for dataset validation the scope is all records).</p><p>You can reference the values from the</p></div>	

```
<ac:
parameter ac:name="
fieldName">details</ac:
parameter>
<ac:
parameter ac:name="
fieldLabel">Details</ac:
parameter>
<ac:
parameter ac:name="type"
>textarea</ac:parameter>
</ac:
structured-macro>
</p>
<p>
<ac:structured-
macro ac:macro-id="
602a7b0b-1968-4c3a-b9f3-
89a41692c486" ac:name="
confi-form-field-
definition" ac:schema-
version="1">
<ac:
parameter ac:name="
fieldName">options</ac:
parameter>
<ac:
parameter ac:name="
fieldLabel">Choose an
option</ac:parameter>
<ac:
parameter ac:name="values"
>false[1=Option 1|2=Option
2|3=Option 3|4=Let me
provide my option|]</ac:
parameter>
<ac:
parameter ac:name="type"
>select</ac:parameter>
</ac:
structured-macro> </p>
<p>
<ac:structured-
macro ac:macro-id="
0ff45634-8fd7-4f78-a2ef-
05e15bb033d8" ac:name="
confi-form-field-
definition" ac:schema-
version="1">
<ac:
parameter ac:name="
fieldName">userOption</ac:
parameter>
<ac:
parameter ac:name="
fieldLabel">My option</ac:
parameter>
<ac:
parameter ac:name="type"
>text</ac:parameter>
</ac:
structured-macro>
</p>
<p>
<br/>
</p>
<p>// this rule
will show the container by
its "ID" when
"showDetails" field is
checked</p>
```

## Action to execute \*

Show container



Choose the action type to perform, see [more details on each action type in our documentation](#)

## Container HTML element

#mycontainer

CSS selector (or selectors as comma separated list) for a "container" HTML element that shall be hidden/shown.

- ☒ With reverse rule (for 'Set field readonly' rule checking this option will unset the readonly state)

When checked, ConfiForms will try to create a reverse rule for given rule (works with Show/Hide field/container, 'Set field readonly' and 'Validate if exists in other Form' rules ONLY)

Conditions are written as filters, more on ConfiForms filters you can find here: [ConfiForms Filters](#) and [ConfiForms Filters by example](#)

```
<p>
  <ac:structured-
macro ac:macro-id="
d8e55975-bb94-49cb-b0e4-
299efb76af22" ac:name="
conform-field-definition-
rules" ac:schema-version="
1">
    <ac:
parameter ac:name="
condition">showDetails:
true</ac:parameter>
    <ac:
parameter ac:name="action"
>Show container</ac:
parameter>
    <ac:
parameter ac:name="
actionFieldName"
>showDetails</ac:parameter>
    <ac:
parameter ac:name="values2"
>#mycontainer</ac:
parameter>
    <ac:
parameter ac:name="
withReverseRule">true</ac:
parameter>
  </ac:
structured-macro>
</p>
<p>// this rule
shows an extra field when
you select an "other"
option in the "<em>options<
/em>" dropdown field (and
hides it otherwise)</p>
<p>
  <ac:structured-
macro ac:macro-id="
7b9d37e2-7f3a-4dff-a7b3-
d88d3402b3b4" ac:name="
conform-field-definition-
rules" ac:schema-version="
1">
    <ac:
parameter ac:name="
condition">options:4</ac:
parameter>
    <ac:
parameter ac:name="
fieldName">userOption</ac:
parameter>
    <ac:
parameter ac:name="action"
>Show field</ac:parameter>
    <ac:
parameter ac:name="
actionFieldName">options<
/ac:parameter>
    <ac:
parameter ac:name="
withReverseRule">true</ac:
parameter>
  </ac:
structured-macro>
</p>
<p>
  <br/>
</p>
</ac:rich-text-
```

```
body>  
    </ac:structured-  
macro>
```

To import via Atlassian [Confluence](#)  
[Source Editor](#)