Link to article: https://discover.claromentis.com/knowledgebase/articles/439

Infocapture: Field Visibility

Field Visibility sets are for controlling **when parts of your form should show and hide,** and **when they are editable**.

Different rules can be set per project role.

This is achieved by grouping the fields you wish to show or hide together in a set and then applying field conditions to determine who can see or edit them and when.

Read on to see an example field visibility set explained in this guide, two more can be found at the links below:

Example 2

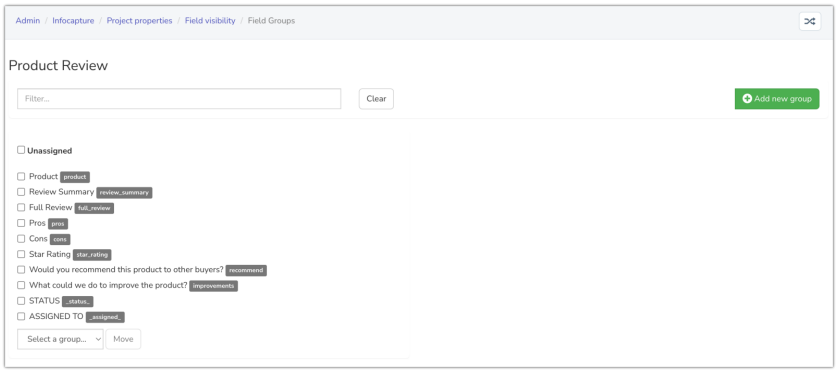
Example 3

Defining Field Groups

Rather than setting rules for each individual field on the form, they are grouped together into a set.

Go to Admin > Infocapture > *Your Project* > Field Visibility > Manage Groups.

All the fields created will be listed and currently in one group called 'Unassigned':



Click 'Add new group' and give the new group a name e.g. *Manager Comment Fields*

Now look for the fields you want to include in the 'Manager Comment Fields' group.

Check the box next to those, select 'Manager Comment Fields' group from the list, and click 'Move'.

Finally, click the 'Save' button at the top of the page, otherwise, the group creation and movement of fields into it will not apply.

Creating a rule

At least one field condition needs to be in place to choose when your field(s) should show and hide.

In the Field Visibility tab, there is now a table for the group you have created, although it doesn't contain any rules yet.

Field visibility rules are defined using at least one field condition, and it's possible to specify what each project role should be able to see and whether they can edit the fields.

As an example:

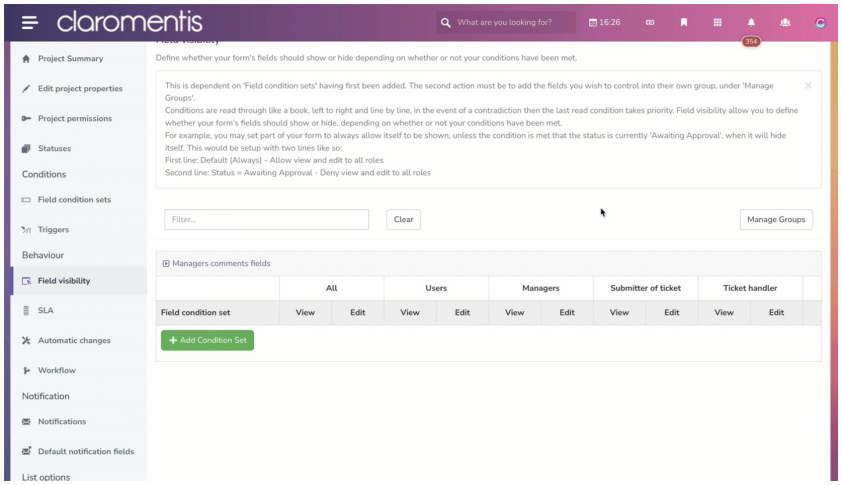
I am using the field condition *Default (always).* This is a hardcoded field condition that is always true.

It's especially useful for field visibility rules and will likely be frequently used across visibility sets in your forms too.

I want my visibility rule to be active at all times so will use Default (Always):

Under the 'All' column, I click on 'view' and choose 'allow'.

Again under the 'All' column, I click 'edit' and choose 'allow'.



Both the project roles (Users and Managers), as well as the hardcoded roles (Submitter and Ticket handler), can all see and edit the fields within the visibility group.

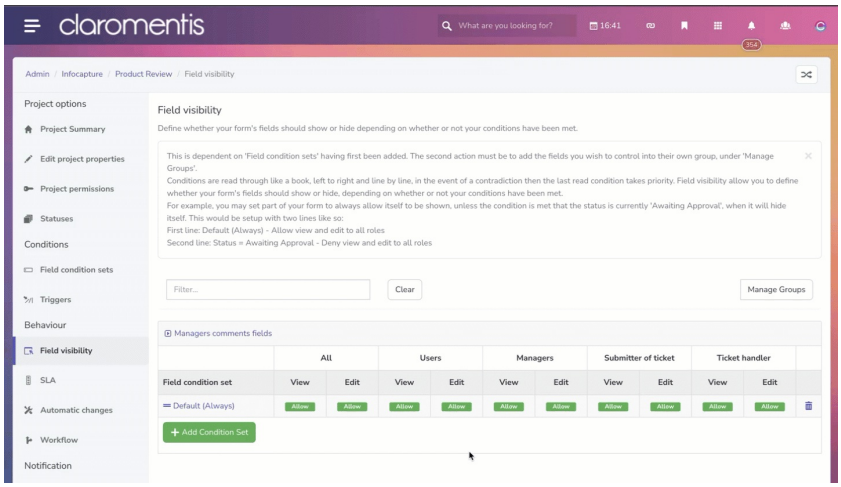
However - this rule is essentially useless!

The behaviour will be the exact same even without this rule because restrictions haven't been applied yet.

Understanding Rules

Now I will make this change, I've clicked into the 'Users' column, and chosen 'Deny' for view and edit.

Notice that the 'All' column automatically changed to 'Mixed' to indicate that there are differences in the permission between roles.



What my rule now means is that anybody in the 'Users' role wouldn't be able to see the fields included in the group, only the 'Managers' role can.

Unless they are the individual who submitted the ticket, or they are the one to whom the form is currently assigned, in which case they can due to the 'Allow' rules set for the 'Submitter' and 'Ticket Handler' roles.

If it's decided the extra permission to allow should not be given if they're the submitter or ticket handler, this can be changed as below:

The view and edit rights for the submitter and ticket handler have been updated to *Not Set.*

'Not Set' means that it won't make any changes to the permissions already been applied.

In this case - if the user is the submitter or the ticket handler, don't make any changes to what's already being decided by what role they're in, 'Users' or 'Managers'.

What I've also changed is to allow 'view' rights to 'Users' for the fields, but 'edit' has been left as 'deny'.

The effect of this rule is that:

- Users within the 'User' project role can the fields, but can't edit them.
- 'Managers' can both see and edit the fields.
- The submitter or ticket handler is not granted any extra permissions, so the rules that will be applied instead depend on whether they're in the 'User' or 'Managers' project role.

How the system applies the Rules

An important rule in understanding field visibility is that **it reads through permissions rules like a book: left to right, line by line. If there's a conflict, it will go with whatever rule it last processed.**

This is why I had to change Submitter and Handler to *Not Set*.

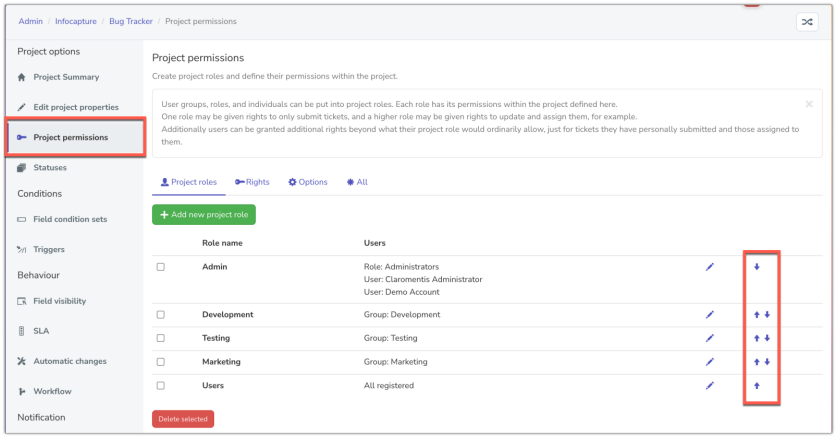
Before I did so, those rules were being read after it had processed the view/edit permissions for the 'Users' role.

So although I'd denied 'Users' permission, it was then allowed again by a later rule that said they were allowed (if the submitter or ticket handler).

When you are making your own field visibility sets keep this in mind when testing as conflict is a likely cause for any difference in behaviour.

Check what roles a user is in and then how the system would process this due to the conflict, is the role processed last taking precedence?

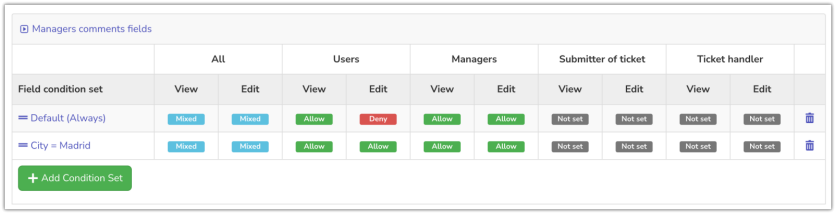
In some cases, you may need to re-order your project roles to ensure the appropriate ones are processed last and the rules you want are applied:



Applying more rules

Now I want to take my rule further. I want it to be the case that *if* a specific city in the UK is selected in the form, *then I do want* 'Users' to be able to edit that field.

I create a field condition for when 'City = Madrid' and edit the field visibility table to include it, then set the rules per project role.



'Users' are allowed to edit when this field condition is in effect.

As this Field Condition comes *after* the Default (Always) condition, it is being processed afterwards and has 'priority' so is being applied, even though the default always rule specifies 'Users' can't edit the field.

But only in situations when 'City = Madrid' is true - when the condition is not met, it will not be true and therefore not in effect, so the Default (Always) condition is applied by the system instead, preventing 'Users' from editing the fields but allowing them to view them.

To quickly rearrange field conditions grab the double bar icon on the far left to drag and drop them, then remember to click 'save' at the top of the screen to apply changes:

However, give thought to your form's logic before rearranging as the order conditions are processed is paramount to how the rules are applied and can be very easily confused, needing an investigation to pick it apart.

e.g. What would happen if I swapped the two example conditions around? It would not work!

In the event that my 'City = Madrid' condition was true, it would be superseded by the Default (Always) condition as it was processed afterwards.

Remember: The Default (always) condition must always go at the top of the table

If it appears anywhere other than the top, it will supersede any other condition as it is always true.

Avoid complexity

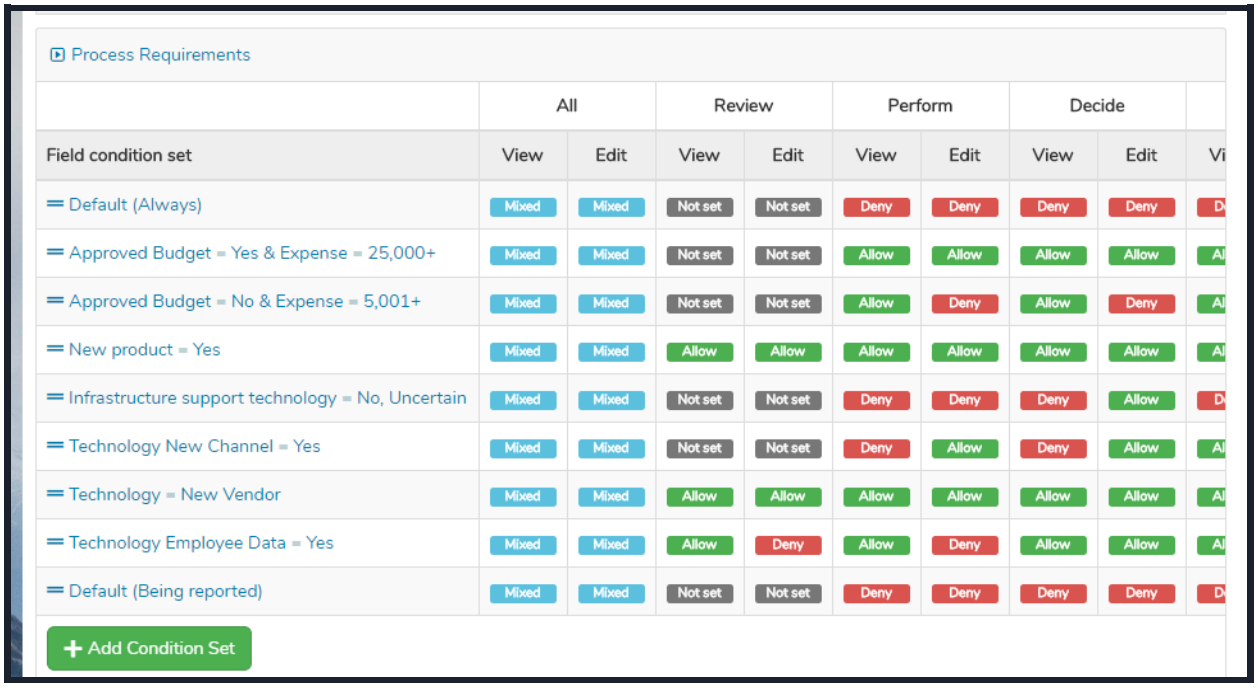
Field permissions are incredibly useful when used well.

Brainstorm with your form management team about how best to set up your own field conditions and visibility sets for your form, then test this rigorously before making it live for end users to submit tickets in.

Follow our top tips to guide your team's form-building process

Don't forget - keep everything simple!

Too many roles and too many roles are cumbersome and often inefficient. Here's an example of a Field Visibility rule that may be unnecessarily complex:



What can't be seen in this image is that the table extends to the right for a further 5 project roles, as well as the submitter and handler columns.

In this instance, there were 9 project roles and hundreds of field visibility sets.

After reviewing, it was found that a huge number of rules and roles were so similar or identical, that they were unnecessary.

The roles, field conditions, and field visibility rules were able to be reduced greatly in number.

Imagine trying to work out why a particular user cannot edit a field when you believe they should and the field visibility set concerned looks like this!

Save your team time troubleshooting and a headache of a logic problem.

Keep your rules simple and ongoing management of the form will be too!

FAQ

I can't see a table to add rules into
The field(s) needs to be in a Field Visibility Group. Go to 'Manage Groups' and make sure it's in a group, and save. You'll then see a table to start adding rules.

The page doesn't change when I make the selection on the form that needs to reveal my hidden fields
Checkout the form, and edit that field. Ensure you've checked *Reload form on changing.* You need the form to reload itself so that it can check through all the Condition rules, and apply any subsequent visibility rules.

I've checked both of the above but my rule doesn't work, the field is always hidden
Have you thought about the order of your Field Conditions in the table? As described in the 'City = Madrid' example above, sometimes one field condition is in effect more often than another, and is superseding the other.
If your logical rule is trying to say *Hide field if X, unless Y.....* then your field condition set for X needs to come first and Y needs to come second.

I'm still not clear. Do you have another example to help me understand?
Try this tutorial here for another example.

Recommended next article:
Automatic Changes

Created on 9 August 2018 by Hannah Door. Last modified on 21 June 2023
Tags: infocapture, user guide, field permissions, field visibility, hidden fields, hidden