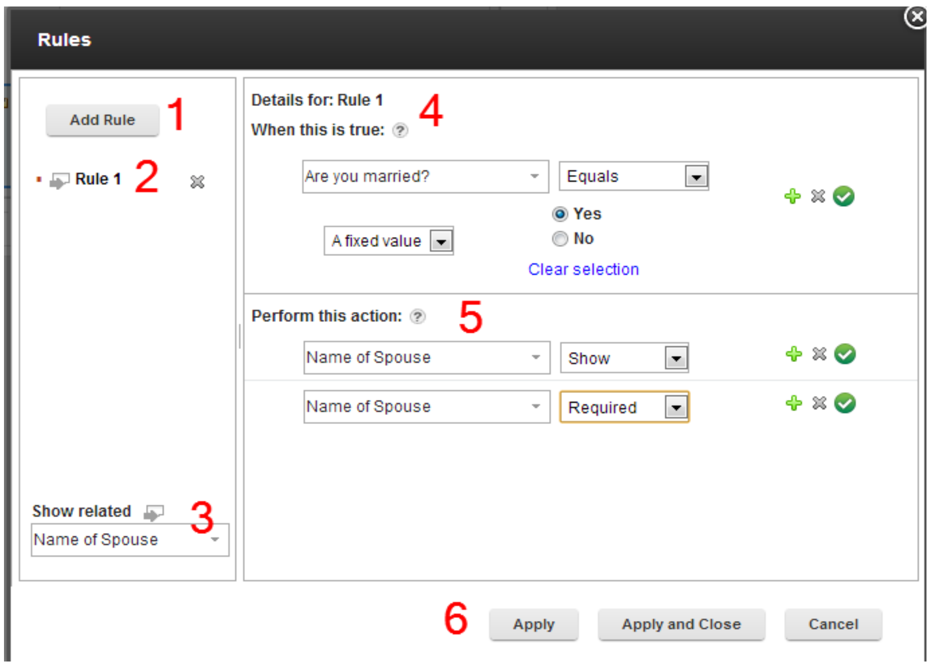
Forms Experience Builder (FEB): Rules Wizard

Version 8.5/8.6

Table of Contents

A detailed look at the FEB Rules Wizard



Details for: Rule 1

When this is true:

Are you married? | Equals

A fixed value | [✓ Yes] No

Perform this action:

Name of Spouse | Show

Name of Spouse | Required

Show related: Name of Spouse

1. If you want to add a new rule you must first click the **Add Rule** button. When the wizard loads it will default to the first rule in the list and if you make changes then you are changing that rule!

2. All the rules in the form will be listed here.
   - A rule can be deleted by clicking the "x"
   - In FEB 8.5/8.6 rules cannot be renamed

3. When you select an item in the **Show Related** dropdown then all the rules that reference or affect this item will be denoted by the symbol in the diagram.
   - The icon (field with gray arrow) to the left of the rule name is tied to the value of the **Show Related** dropdown.

4. Every rule starts with a condition that must be met.
   - The first dropdown contains all the form fields that can be used in a condition
   - The second dropdown contains the valid operators which change depending on the datatype of the item you select in the first dropdown.
   - The third dropdown changes the reference from a fixed value to the value of another item in the form.
   - you can add more conditions by clicking the "+"
   - you can only have one type of relationship between the conditions (either "AND" or "OR")

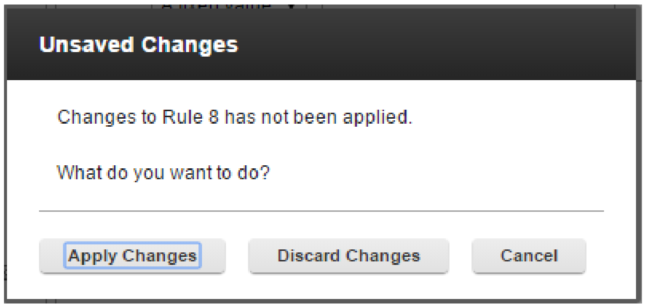
5. Each Rule can perform multiple actions; hide/show, enable/disable, required/not required, valid/not valid.

Note: When you define an action then the inverse is also true. For example in the above screenshot, if married equals "yes" then name of spouse will be shown which also means if married is "no" or not selected then name of spouse will be hidden.

As you create rules it is important to remember that rules can affect each other and in some cases completely counteract what you have defined. You should never refer to the same item in the "Perform this action" section for multiple rules. The general rule here is that if there are conflicts then the "last" rule wins.

6. **Apply** will create the rule but leave the dialog open
Apply and Close will create the rule and close the dialog.
Cancel will close the dialog and it will not keep any changes made since the last time "Apply" was clicked or the dialog was opened.

Note: If you navigate to a different rule, before applying your changes you will see a dialog:


Unsaved Changes

Changes to Rule 8 has not been applied.

What do you want to do?

If you click "Apply Changes" then your rule modifications will be saved. If you click "Disgard Changes" then any changes made will be thrown away. If you click "Cancel" then the attempt to navigate away from the current rule will be canceled.

Operators Explained

Number/Currency Field

Between	Checks if the selected field value is between either two fixed values or other form items.
Does not equal	
Equals	
Greater than	
Greater than or equals	
Less than	
Less than or equals	

SingleLine Entry / Email / Multi-line Entry

Between	Checks if the value of the selected field falls alphabetically between either two fixed values or other form items. i.e. "Red" is between "Green" and "Yellow" because "R" is between "G" and "Y" in the alphabet. Case matters here, as the string is evaluated against the alphabet by first scanning the uppercase letters (A-Z) and then the lowercase (a-z). For example if you specified between "Green" and "brown", then "Red" would be valid but "red" would not, because the alphabet range would be A-Z + a-b.
Contains	Does the specified string contain the fixed value or other field value. This function recognizes case.
Ends With	Does the specified string end with the fixed value or other field value. This function recognizes case.
Equals	Does the specified string equal the fixed value or other field value. Case matters, "c" does not equal "C".
Follows	Does the specified string come after the fixed value or other field value in the alphabet. i.e. "Red" follows "Green". This function recognizes case.
Matches	Does the specified string match after the fixed value or other field value. Similar to contains, it is true if the specified string can be found anywhere in the value. This function recognizes case.
Precedes	Does the specified string come before the fixed value or other field value in the alphabet. i.e. "Red" precedes "Yellow". This function recognizes case.
Starts with	Does the specified string start with the fixed value or other field value. This function recognizes case.

Select One / Checkbox / Dropdown

Equals	This function recognizes case.
Does not equal	This function recognizes case.

Select Many

Matches	This function recognizes case.
Does not match	This function recognizes case.
Includes	This function recognizes case.
Does not include	This function recognizes case.

Time / Timestamp

After	
Before	
Between	
Equals	

Date

After	Returns true if the date specified is “after” the exact value or the value of the other form item specified.
Before	Returns true if the date specified is “before” the exact value or the value of the other form item specified.
Between	
Day is	Compares the day of the date specified with an exact value or the value of another item.
Equals	
Month is	Compares the month of the date specified with an exact value or the value of another item.
Year is	Compares the year of the date specified with an exact value or the value of another item.

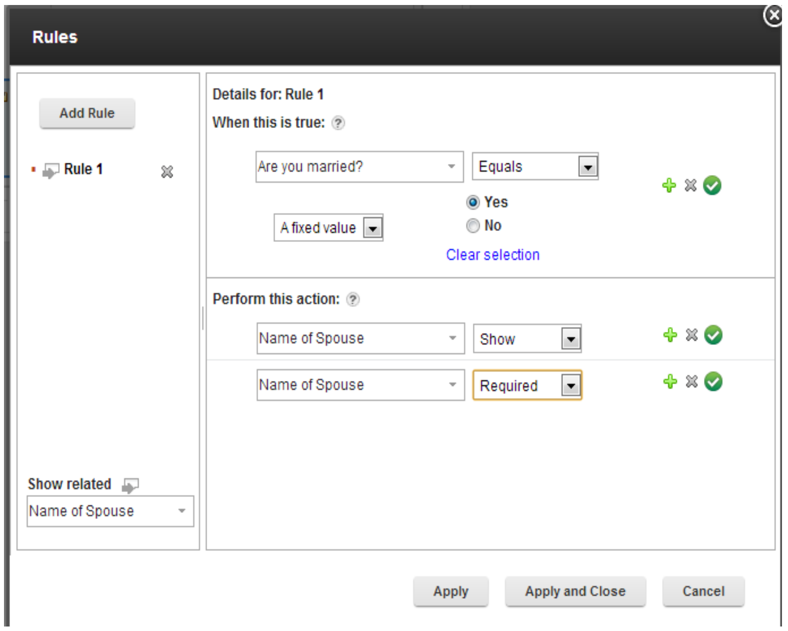
Actions Explained

Show	Shows the specified fields

Hide	Hides the specified fields
Enable	Enables the specified fields
Disbale	Disables the specified fields, they will appear as light gray and cannot be interacted with.
Valid	Sets the field to valid.
Not Valid	Sets the field to invalid, a red message will appear under the field and a exclamation will appear before the field label.
Required	Sets the field to required. The form will not submit unless the value is entered. User will receive a warning if they try to navigate to a different page without filling in the value.
Not Required	Sets the field to not required (optional).

Simple Rule 1 – Radio Selection

This rule will make the Name of Spouse field visible and required if the user selects “Yes” to the Are you married question.



Rules

Details for: Rule 1

When this is true:

Are you married? Equals

A fixed value: (●) Yes ( ) No

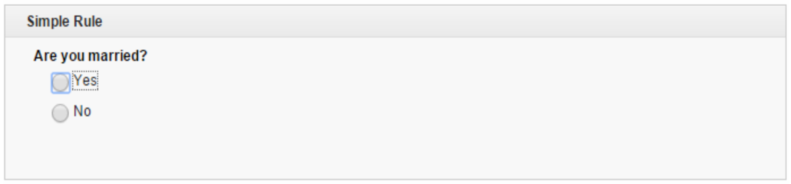
Perform this action:

Name of Spouse Show

Name of Spouse Required

Show related: Name of Spouse

When you first render the form the **Name of Spouse** field is not shown:



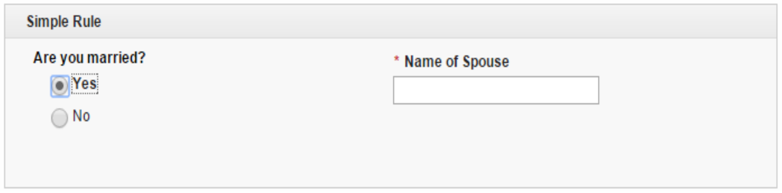
Simple Rule

Are you married?

( ) Yes

( ) No

When you select **Yes** the **Name of Spouse** field appears and becomes required:



Simple Rule

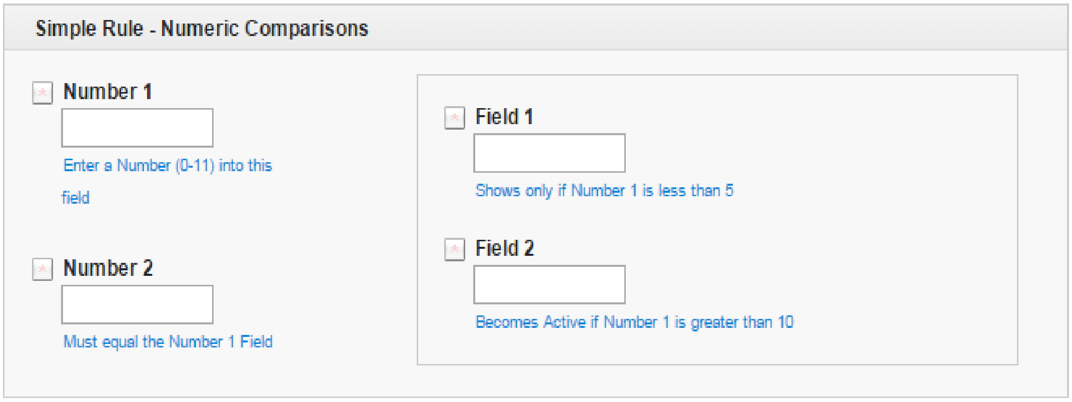
Are you married?

(●) Yes

( ) No

* Name of Spouse

Simple Rule 2 – Numeric Comparisons


Simple Rule - Numeric Comparisons

Number 1

Enter a Number (0-11) into this field

Number 2

Must equal the Number 1 Field

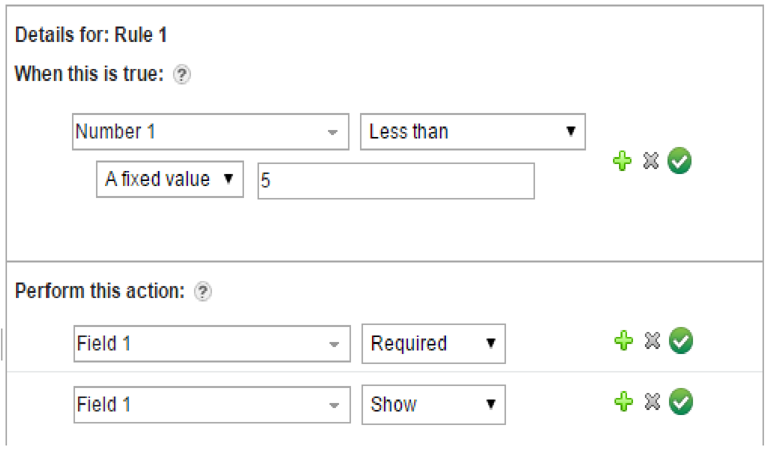
Field 1

Shows only if Number 1 is less than 5

Field 2

Becomes Active if Number 1 is greater than 10

We will look at a few different rules in this section. The first rule will cause **Field 1** to be shown and set to required:


Details for: Rule 1

When this is true:

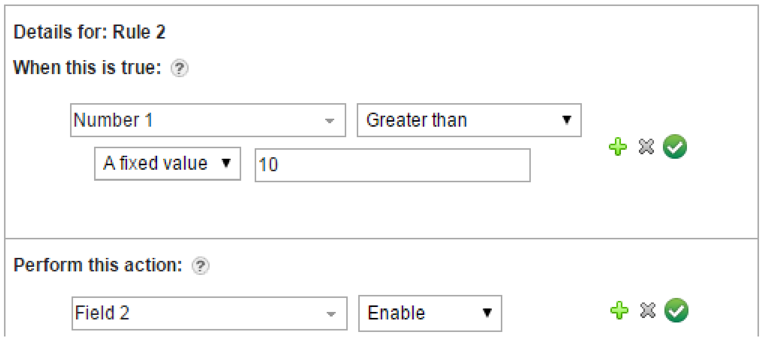
Number 1 | Less than | A fixed value | 5

Perform this action:

Field 1 | Required

Field 1 | Show

The second rule will cause **Field 2** to be enabled for user input:


Details for: Rule 2

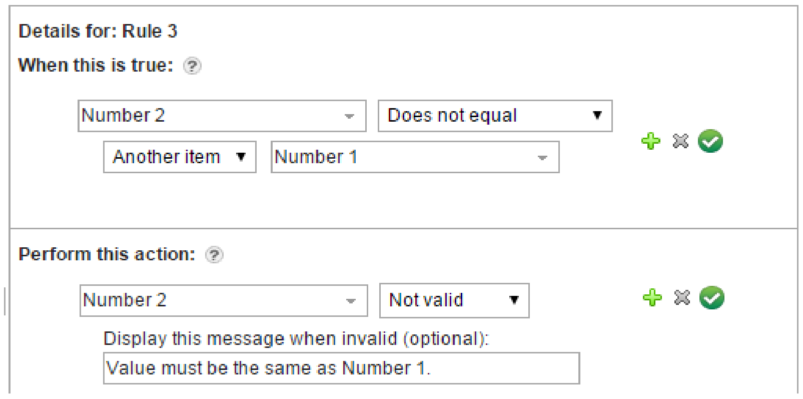
When this is true:

Number 1 | Greater than | A fixed value | 10

Perform this action:

Field 2 | Enable

The third rule will set **Number 2** to invalid if its value does not match what is entered in **Number 1**:


Details for: Rule 3

When this is true:

Number 2 | Does not equal

Another item | Number 1

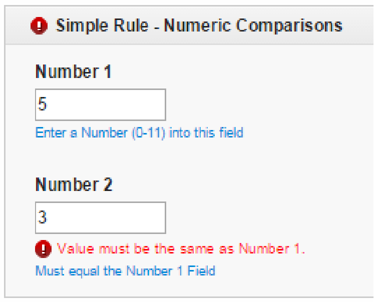
Perform this action:

Number 2 | Not valid

Display this message when invalid (optional):

Value must be the same as Number 1.

Note that this rule uses the "Another item" selection that allows you to compare one field to another on the same form. In the "Perform this action" section you can also see that we can set the field to invalid and set the message that will be displayed when the condition is met:


Simple Rule - Numeric Comparisons

Number 1

5

Enter a Number (0-11) into this field

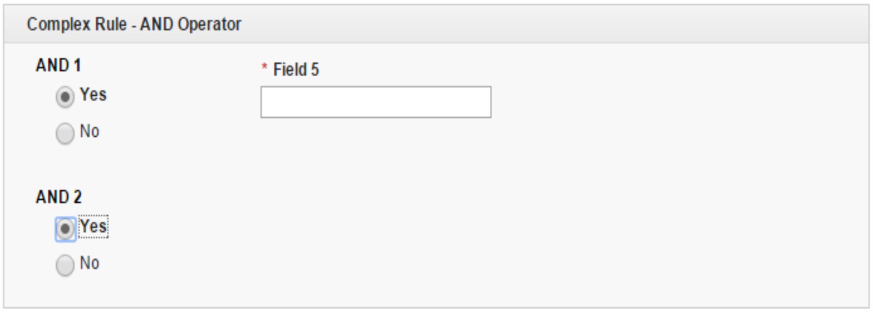
Number 2

3

Value must be the same as Number 1.

Must equal the Number 1 Field

Rule with an AND Condition



Complex Rule - AND Operator

AND 1

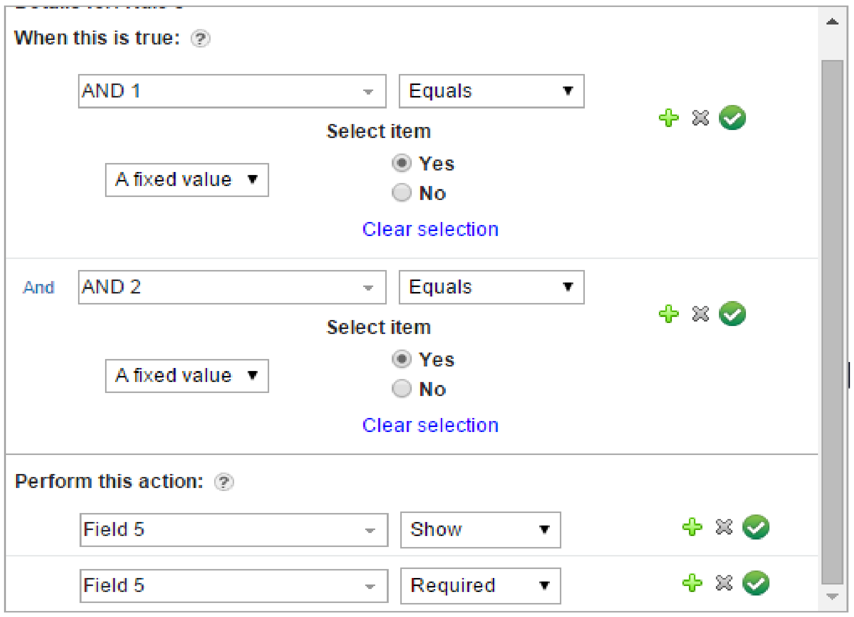
[✓ Yes] No

AND 2

[✓ Yes] No

* Field 5

In this section we look at creating a rule where two fields are part of the condition. If the value of both **AND 1** and **AND 2** is "Yes" then show **Field 5**.



When this is true:

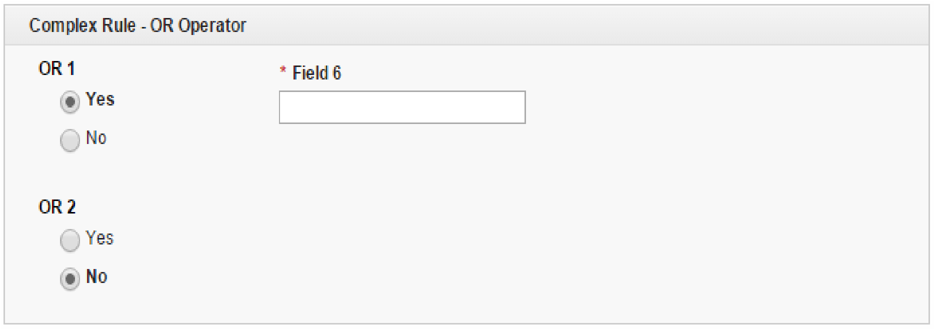
	AND 1	Equals	
	A fixed value	Select item: [✓ Yes] No	Clear selection
And	AND 2	Equals	
	A fixed value	Select item: [✓ Yes] No	Clear selection

Perform this action:

Field 5	Show
Field 5	Required

Note that you can only have ONE relationship when you have more than one field in the condition. If you need to have more complicated logic then you will have to refer to that section further in the document.

Rule with an OR Condition



Complex Rule - OR Operator

OR 1

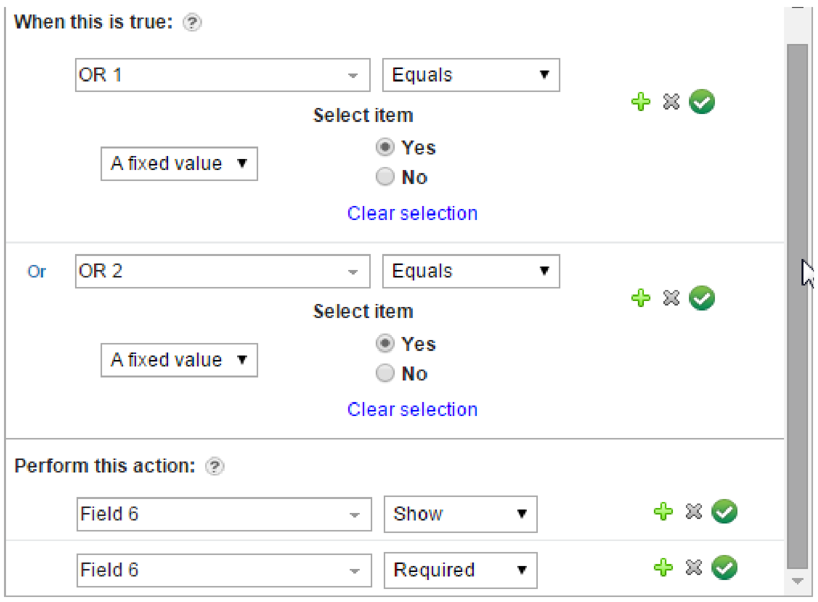
[✓ Yes] No

OR 2

Yes [✓ No]

* Field 6

In this section we look at creating a rule where two fields are part of the condition. If **OR 1** or **OR 2** equal “Yes” then we will show **Field 6** and make it required.


When this is true:

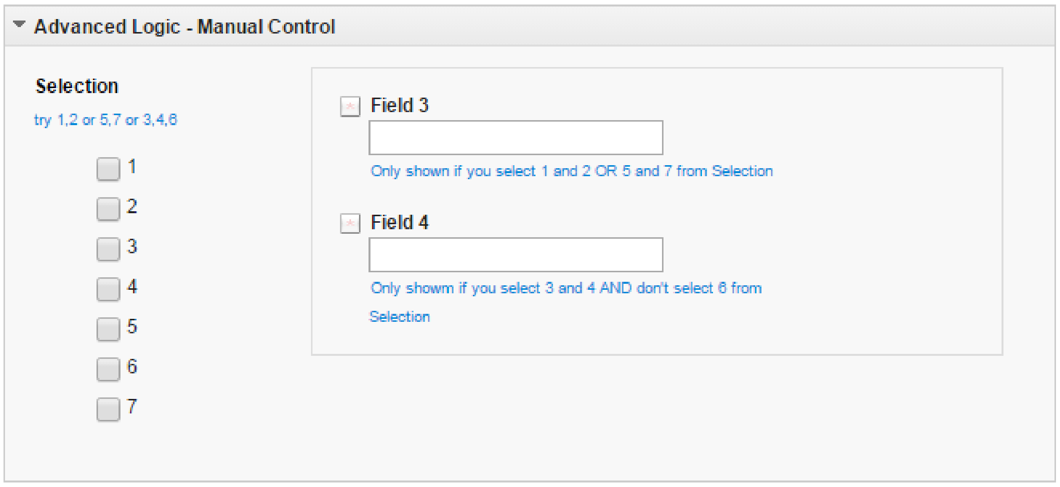
	OR 1	Equals	
	A fixed value	Select item: (● Yes ○ No) Clear selection	
Or	OR 2	Equals	
	A fixed value	Select item: (● Yes ○ No) Clear selection	

Perform this action:

Field 6	Show
Field 6	Required

How to Handle More Complicated Logic

The Rules wizard does have its limitations and you may find that you have certain scenarios where you need additional flexibility. You can create your own customized behavior by using JavaScript. Let's review what that might look like.


Advanced Logic - Manual Control

Selection

try 1,2 or 5,7 or 3,4,6

☐ 1
☐ 2
☐ 3
☐ 4
☐ 5
☐ 6
☐ 7

Field 3

Only shown if you select 1 and 2 OR 5 and 7 from Selection

Field 4

Only showm if you select 3 and 4 AND don't select 6 from Selection

Let's say that we want to establish a rule where a field will only appear if ONLY 1 and 2 are selected or ONLY 5 and 7 are selected (but not both). I would write a JavaScript function like this:

```
app.getSharedData().showField1 = function() {
  var v = BO.F_SelectMany.getValue();
  if( (v === "1__#__2") || (v === "5__#__7")) {
    form.getPage('P_NewPage').F_SingleLine4.setVisible(true);
  } else {
    form.getPage('P_NewPage').F_SingleLine4.setVisible(false);
  }
}
```

Within JavaScript we can leverage its full power and therefore create any complex comparison. There are a few things that you need to know to get started creating your own logic:

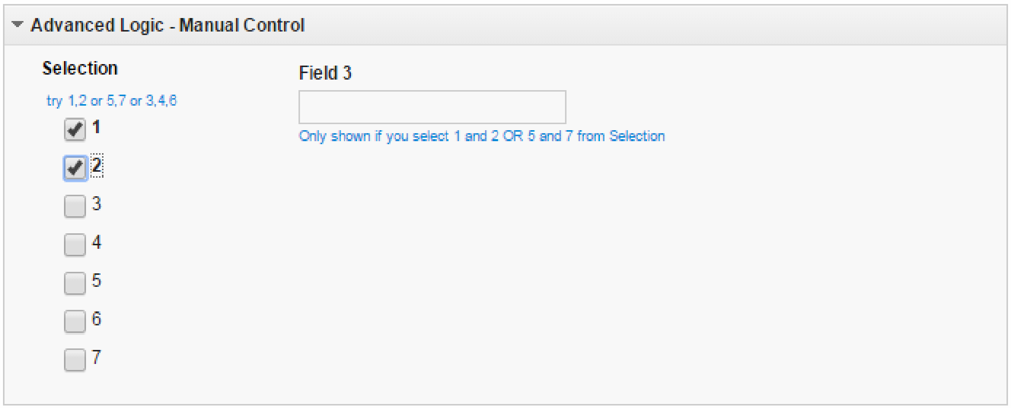
1. The AND operator in JavaScript is &&
2. The OR Operator in JavaScript is ||
3. The value of a Checklist is a list of the selected items separated by "__#__" (that is underscore underscore number sign underscore underscore).
4. You can affect an item's behavior by using the functions; setVisible(), setRequired(), setActive(). In all cases you pass either true or false.

To use this function, I add the following to the Form **onLoad** event as well as the **onItemChange** event of the item that the user interacts with (in this case it is the Checklist):

```
    app.getSharedData().showField1();
```

The purpose for calling this function in multiple places is that we need to make sure that we evaluate the state of the field when the form loads and again when the conditional item changes.

Here you see the function in action:



Advanced Logic - Manual Control

Selection

try 1,2 or 5,7 or 3,4,6

- [x] 1
- [x] 2
- [ ] 3
- [ ] 4
- [ ] 5
- [ ] 6
- [ ] 7

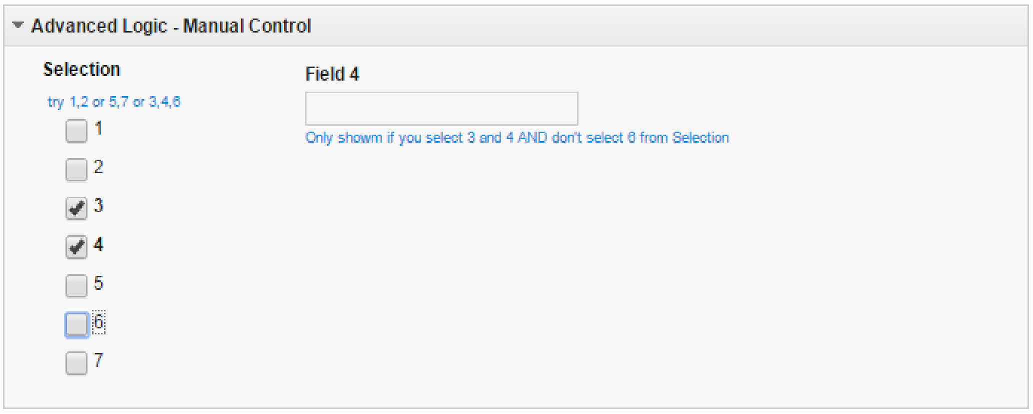
Field 3

Only shown if you select 1 and 2 OR 5 and 7 from Selection

Let's say you want to have 1 and 2 or 5 and 7 (where all 4 selected is also valid), then you would modify the function to:

```
    app.getSharedData().showField1 = function() {
      var v = BO.F_SelectMany.getValue();
      if((v.indexOf("1") !== -1 && v.indexOf("2") !== -1) ||
    (v.indexOf("5") !== -1 && v.indexOf("7") !== -1)) {
        form.getPage('P_NewPage').F_SingleLine4.setVisible(true);
      } else {
        form.getPage('P_NewPage').F_SingleLine4.setVisible(false);
      }
    }
```

Let's look at another complex rule:


Advanced Logic - Manual Control
Selection
try 1,2 or 5,7 or 3,4,6
1
2
3
4
5
6
7
Field 4
Only showm if you select 3 and 4 AND don't select 6 from Selection

For this one, we will only show **Field 4** if the user selects 3 and 4 and not 6. The function looks like:

```
app.getSharedData().showField2 = function() {
var v = BO.F_SelectMany.getValue();
if( (v.search("3") != -1 && v.search("4") != -1) && (v.search("6")
=== -1)) {
  form.getPage('P_NewPage').F_SingleLine5.setVisible(true);
} else {
  form.getPage('P_NewPage').F_SingleLine5.setVisible(false);
}
}
```

For more information on what functions are available when writing your own JavaScript refer to our Knowledge Center.

Now that we have demonstrated several different ways of creating rules within a FEB application you should be well equipped to start creating dynamic behavior in your forms.

We are constantly evaluating our features and functions in an effort to make them easier to use and feature rich. If you have any recommendations or have encountered any issues please let us know!