ACCESSGOV

Calculations Guide

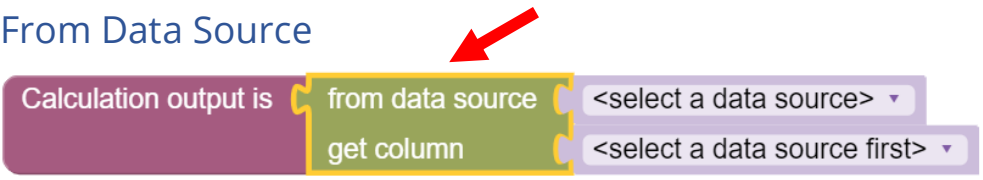
Value

Category: Data & Text
What it does: Returns a constant value that you define.
How to use it: Use the "Value" piece when a constant text or value is desired. Use this piece to provide information to end users, such as the constant text content of an email on submission or a constant value in a mathematical operation.

Form Input

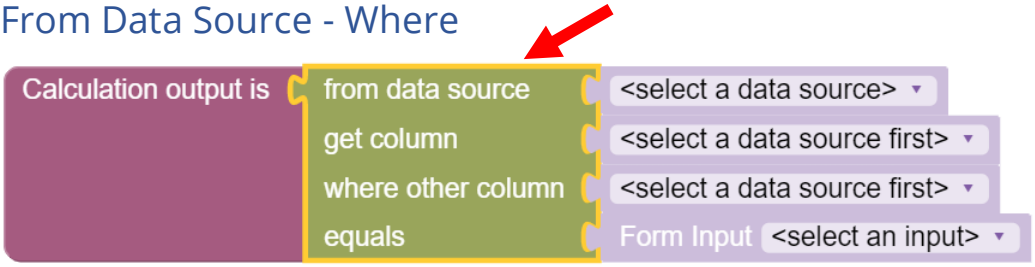
Category: Data & Text
What it does: Returns the value entered by an end user in the selected form input.
How to use it: Use the "Form Input" piece when a user's input is needed, such as the name, email, or phone number of an end user. For example, the returned value can be a personalized email on submission using the user's name or using a user's input to calculate a flexible result, such as price.

From Data Source



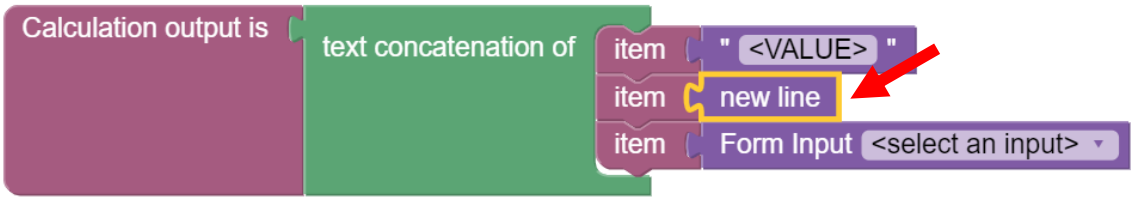
Category: Data & Text
What it does: Returns a list of all values in one column of a data source, including duplicates.
How to use it: Use the "From Data Source" piece to display a list maintained in the Data Sources admin function. For example, use this piece to display a list dynamically based on user input or an informational list of items maintained in one reusable data source to convey information to end users.

From Data Source - Where



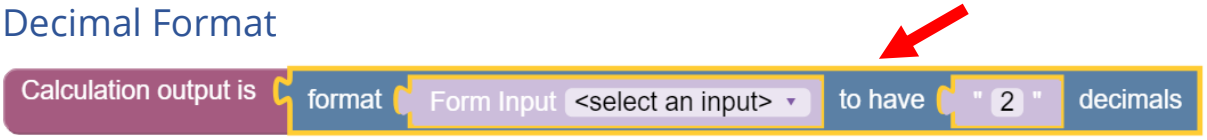
Category: Data & Text
What it does: Returns all values from a data source in Column A, where Column B's value equals the value entered by an end user, including duplicates.
How to use it: The "From Data Source - Where" piece is similar to the From Data Source piece, but it filters out items based on another columns matched criteria, rather than simply returning an entire column. Use this piece to validate user input against a list of values or dynamically generate a list of options based on user input. For example, use this piece to provide information to end users, only if their information is valid; calculate a price based on user input and mapped values in a Data Source; or display meaningful text based on less meaningful values behind the scenes.

New Line



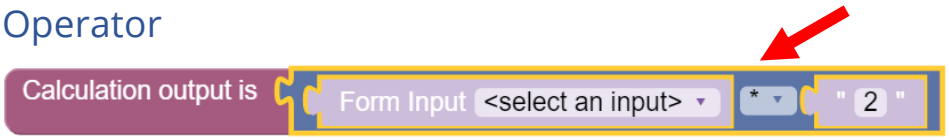
Category: Data & Text
What it does: Creates a new line when joining text together.
How to use it: Use the "New Line" piece within a Concatenation or Concatenation (List) piece to create a new line in text. For example, use this piece when building a large block of text that contains both constant text and values based on user input and has information that should be broken into parts, such as in email body text.

Decimal Format



Category: Data & Text
What it does: Formats a value as a decimal using as many places as you define.
How to use it: Use the "Decimal Format" piece to display numeric values as a decimal. Values are returned with the defined decimal places and rounded as appropriate. For example, use this piece to format a number to two decimal places when returning a dollar amount.

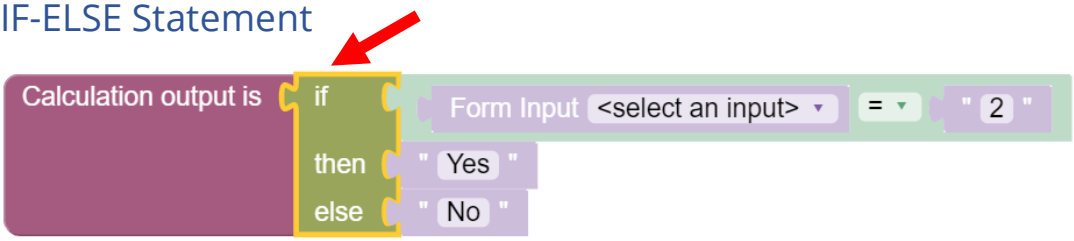
Operator



Category: Operations
What it does: Operates on two provided values to return a mathematical result.
How to use it: Use the "Operator" piece to perform the following operations:

- add (+) – addition of two values – *numeric*
- subtract (-) – subtraction of one value from another value – *numeric*
- multiply (*) – multiplication of two values – *numeric*
- divide (/) – division of one value by another value – *numeric*
- modulo (%) – returns the remainder after dividing the first value by the second value – *numeric*

IF-ELSE Statement

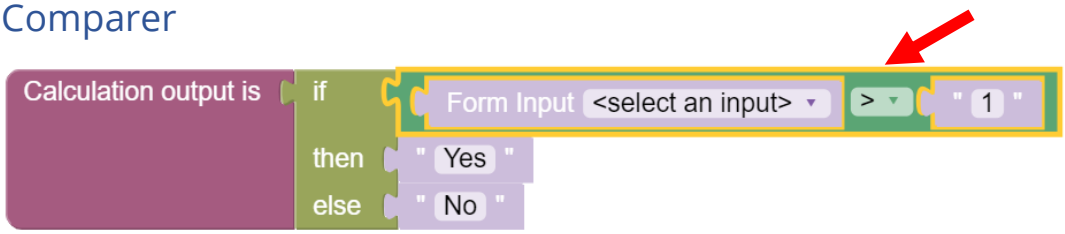


Category: Operations
What it does: Checks provided "if" conditions and returns "then" value if true or "else" value if not true.
How to use it: Provide a condition for "if" using any combination of the following pieces:

- Comparer
- AND/OR
- NOT
- Is Empty

An IF-ELSE Statement can be made by nesting another IF-ELSE Statement in the "else" of the first IF-ELSE Statement block. For example, use this piece to display information to end users based on their input; validate user-input information; ensure dates fall within an appropriate range; dictate whether a user needs to pay a certain amount based on selections; or perform more complex logic to show or hide a conditional region.

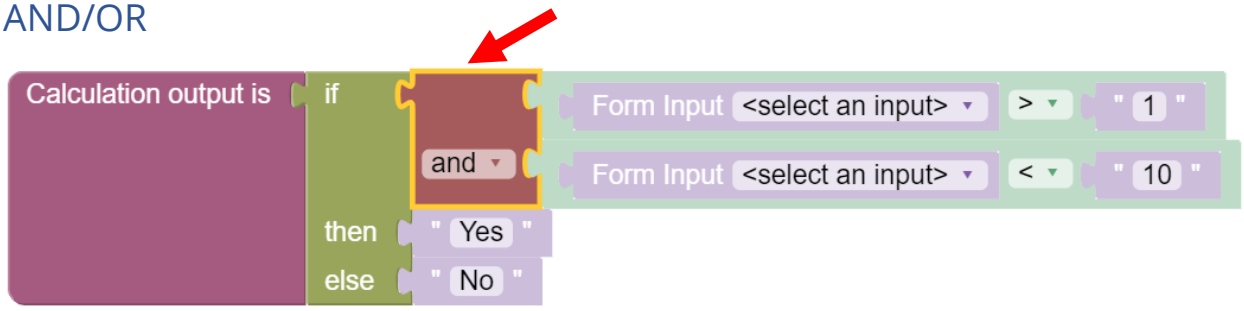
Comparer



Category: Operations
What it does: Compares two provided values and returns "then" value if true or "else" value if not true.
How to use it: Use the "Comparer" piece in an IF-ELSE Statement to perform the following comparisons:

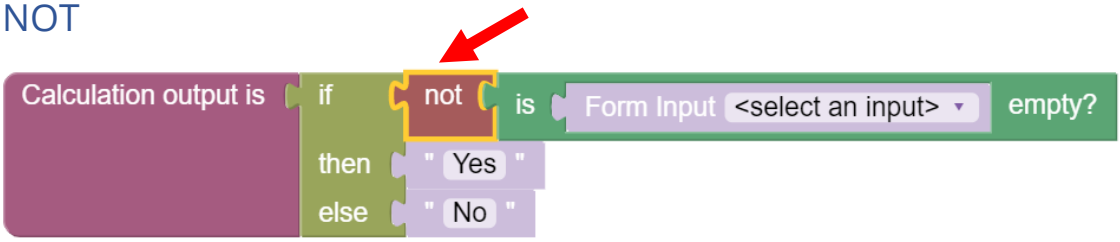
- = – two values are equal – *numeric or alphanumeric*
- ≠ – two values are not equal – *numeric or alphanumeric*
- < – left value is less than right value – *numeric*
- > – left value is greater than right value – *numeric*
- <= – left value is less than or equal to right value – *numeric*
- >= – left value is greater than or equal to right value – *numeric*
- == – two values are equal – *numeric*
- != – two values are not equal – *numeric*
- equals – two values are equal – *alphanumeric*
- not equals – two values are not equal – *alphanumeric*

AND/OR



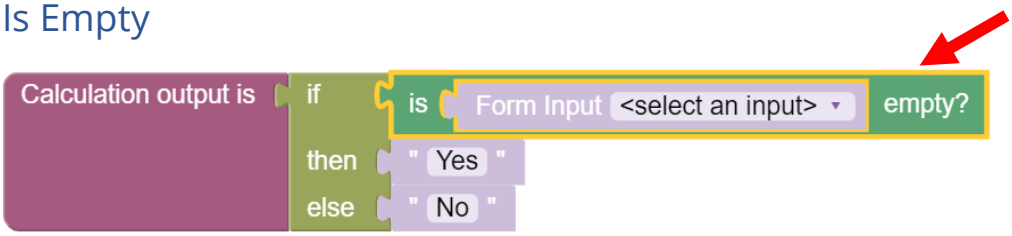
Category: Operations
What it does: Checks if two conditions are both true (AND) or if one condition is true (OR).
How to use it: Use the "AND/OR" piece in an IF-ELSE Statement to check more than one condition at a time. "AND/OR" pieces can be nested within one another to check more than just two conditions. For example, use this piece to check if two user input values fall within a range or if a user has selected at least one checkbox.

NOT



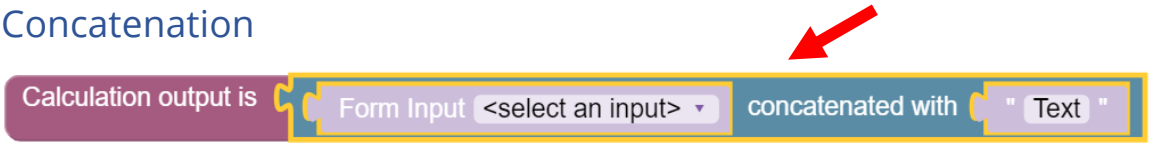
Category: Operations
What it does: Returns the inverse of a provided comparison.
How to use it: Use the "NOT" piece in an IF-ELSE Statement to return the inverse result of a comparison of two values. For example, use this piece to check if a user has entered a value (i.e. is NOT empty) or if multiple conditions are not met.

Is Empty



Category: Operations
What it does: Returns whether an input on the form has an entered or selected value.
How to use it: Use the "Is Empty" piece in an IF-ELSE Statement to check if a user has not entered or selected a value for an input on the form. For example, use this piece to display content to users if they have not entered information into an optional field or check if users have not selected any options in a check or radio group.

Concatenation



Category: Operations
What it does: Joins two alphanumeric values together.
How to use it: Use the "Concatenation" piece to join two alphanumeric values together to create a string. For example, use this piece for very simple concatenations of text, such as joining a dollar sign to a number. If joining together more than two values, use Concatenation (List).

Add to Date

Category: Date
What it does: Adds provided amount of time units to a date.
How to use it: Use the "Add to Date" piece to add defined time units to a constant date or a date provide by an end user. The following units are available:

- Years
- Months
- Days
- Hours
- Minutes
- Seconds

For example, use this piece to add five years to the Current Date to let users know when they must fill out the form again, assuming it must be done again five years after being completed. To use this piece, you must provide a date. Acceptable formats are:

- MM/dd/yyyy
- MM-dd-yyyy
- MM/dd/yyyy hh:mm:ss
- MM-dd-yyyy hh:mm:ss

Date Difference

Category: Date
What it does: Returns the difference in time units between two provided dates.
How to use it: Use the "Date Difference" piece to calculate the difference between two dates measured in one of the following units:

- Years
- Months
- Days
- Hours
- Minutes
- Seconds

For example, use this piece in conjunction with an IF-ELSE Statement to check whether a date entered by a user falls within a certain period or to calculate a user's age using the Current Date. To use this piece, you must provide dates. Acceptable formats are:

- MM/dd/yyyy
- MM-dd-yyyy
- MM/dd/yyyy hh:mm:ss
- MM-dd-yyyy hh:mm:ss

Part of Date

Category: Date
What it does: Returns a specific part of a provided date.
How to use it: Use the "Part of Date" piece to return the following parts of a date:

- Year
- Month
- Day
- Hour
- Minute
- Second

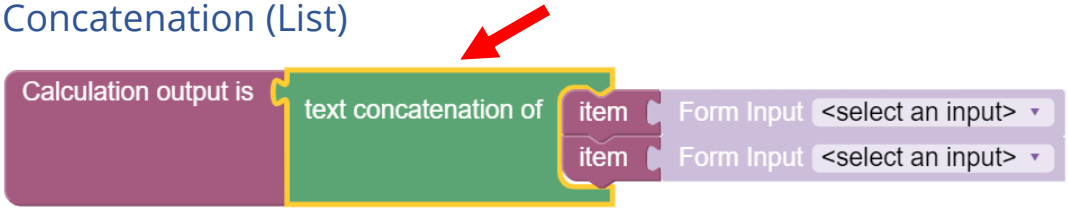
For example, use this piece to return the current year or use it in conjunction with a Concatenation (List) to make a custom date format, such as Month/Day for example. To use this piece, you must provide a date. Acceptable formats are:

- MM/dd/yyyy
- MM-dd-yyyy
- MM/dd/yyyy hh:mm:ss
- MM-dd-yyyy hh:mm:ss

Current Date

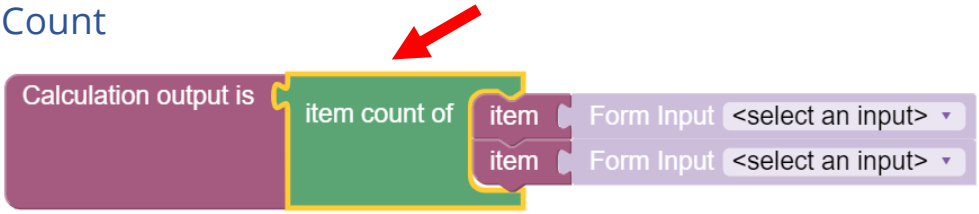
Category: Date
What it does: Returns the current date or date and time in the format of *MM/dd/yyyy hh:mm:ss*.
How to use it: Use the "Current Date" piece to display the current date and time to end users or pass it into another date function. For example, use this piece to display the current date as part of a signature block; use the current date in an Add to Date to calculate a certain time relative to the current day; use the Date Difference to calculate users' age based on input; or pass the current date into a Part of Date to return the current year.

Concatenation (List)



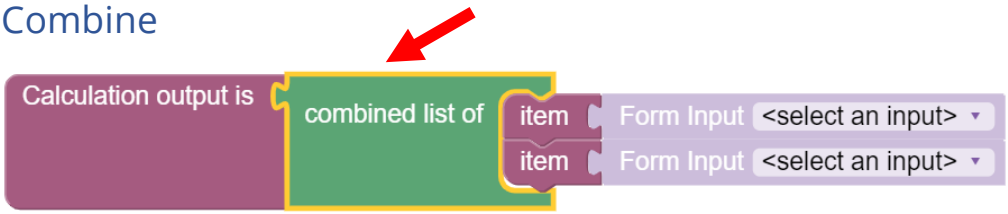
Category: List
What it does: Joins a list of alphanumeric values together.
How to use it: Use the "Concatenation (List)" piece to join a series of alphanumeric values together to create a string. Use the Concatenation piece when joining a small amount of values together, but use this piece to join much larger sets of values into a string. For example, use this piece to compose a personalized email body that contains several Values, New Lines, and Form Input values.

Count



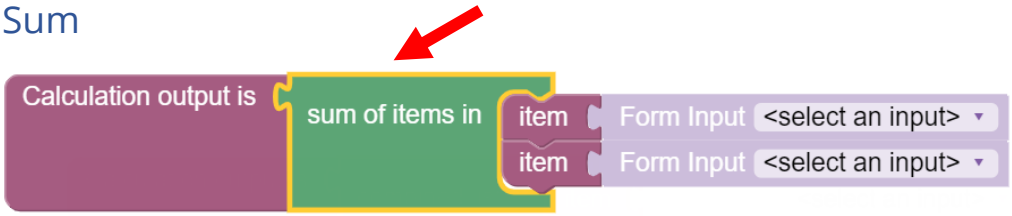
Category: List
What it does: Returns the total count of items in a list.
How to use it: Use the "Count" piece to calculate the number of items selected or entered by a user. It can be used to count the items in a list, the selections in a check group, or the repeatable regions added to a form. For example, use this piece to validate if a user has selected a minimum number of items from a check group.

Combine



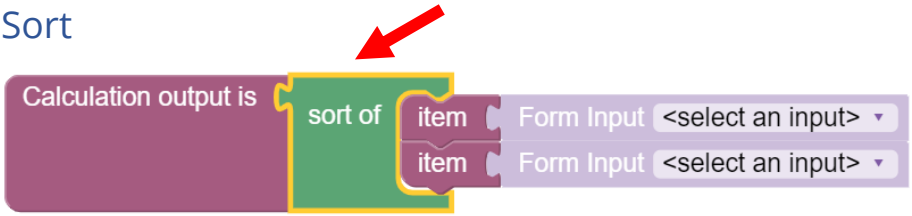
Category: List
What it does: Returns a combined list of provided lists, including duplicates.
How to use it: Use the "Combine" piece to join lists together. For example, use this piece to combine users' selections from separate check groups to display in an email or combine two data source lists that are managed separately to display to the end users. To combine lists without duplicates, see Union.

Sum



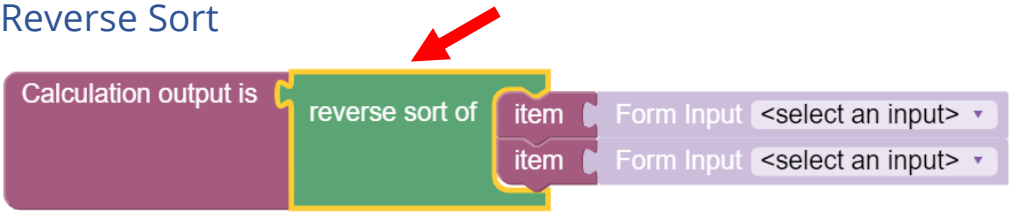
Category: List
What it does: Returns the sum of the values in the provided lists.
How to use it: Use the "Sum" piece to return a running total of multiple items. For example, use this piece to return a total price of the sum of a dollar value in a repeatable region or add a fixed value to a list of user input values.

Sort



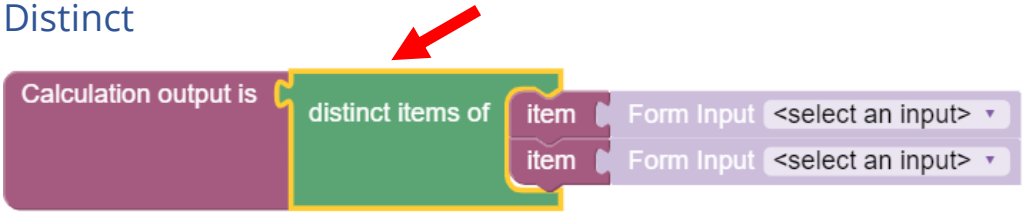
Category: List
What it does: Returns the provided list, sorted A to Z or in ascending order.
How to use it: Use the "Sort" piece to sort lists that are easier for users to read when sorted logically. You can sort numeric and alphanumeric values. For example, use this piece to display to an end user a list of unsorted values from a data source from A to Z. To sort the opposite direction, see Reverse Sort.

Reverse Sort



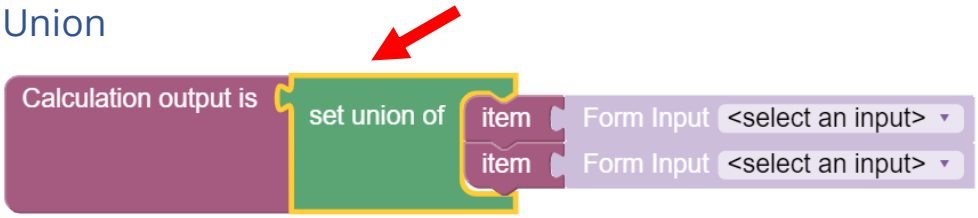
Category: List
What it does: Returns the provided list, sorted Z-A or in descending order.
How to use it: Use the "Reverse Sort" piece to sort lists that are easier for users to read when sorted logically, but in reverse order. You can sort numeric and alphanumeric lists. For example, use this piece to display to an end user a list of unsorted values from a data source in descending order. To sort the opposite direction, see Sort.

Distinct



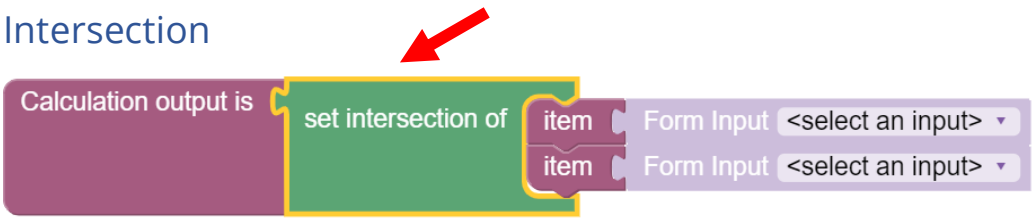
Category: List
What it does: Returns only unique values from a list.
How to use it: Use the "Distinct" piece to strip out duplicate values from a list. The calculation must be performed on a list of items. For example, use this piece to only return distinct categories in a data source column that includes redundant categories.

Union



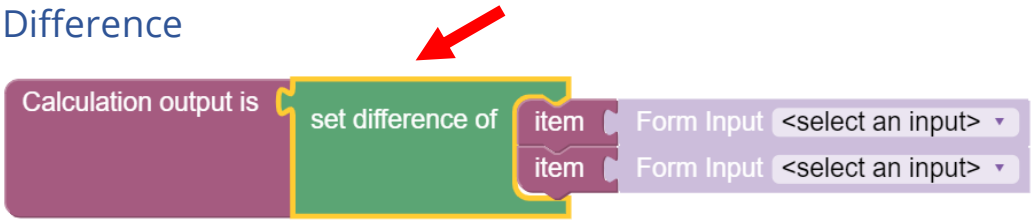
Category: List
What it does: Returns a combined list of all values and eliminates duplicates.
How to use it: Use the "Union" piece to combine lists without duplicate values. You can combine numeric and alphanumeric lists. The calculation must provide at least two lists for the union to occur. For example, use this piece to take multiple lists from data sources and combine them to be used in a dropdown list. To also return duplicate values, see Combine.

Intersection



Category: List
What it does: Returns a combined list of shared values and eliminates duplicates.
How to use it: Use the "Intersection" piece to combine lists without duplicate values. You can combine numeric and alphanumeric lists. The calculation must provide at least two lists for the intersection to occur. For example, use this piece to take multiple lists from data sources and combine them to be used in a dropdown list. To return a list of unshared values, see Difference.

Difference

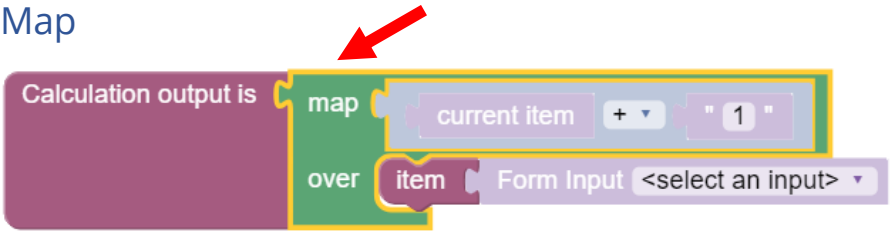


Category: List
What it does: Returns a combined list of values not shared and eliminates duplicates.
How to use it: Use the "Difference" piece to combine lists without duplicate values. You can combine numeric and alphanumeric lists. The calculation must provide at least two lists for the difference to occur. For example, use this piece to Take multiple lists from data sources and combine them to be used in a dropdown list. To return a list of shared values, see Intersection.

Item

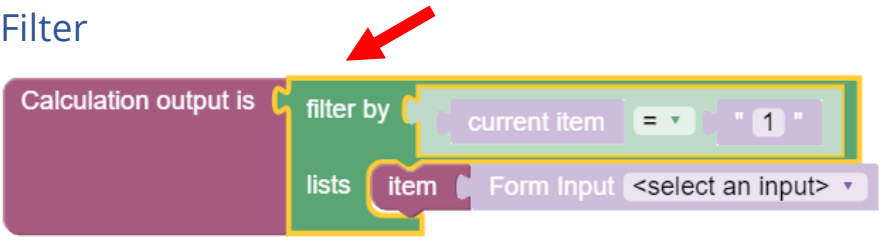
Category: List; Advanced
What it does: Acts as a list item within a list.
How to use it: The "Item" piece must be used within list functions in the List category or the Map, Filter, or Fold functions in the Advanced category. For example, use this piece for every component of text that makes up the concatenation when constructing the body of an email.

Map



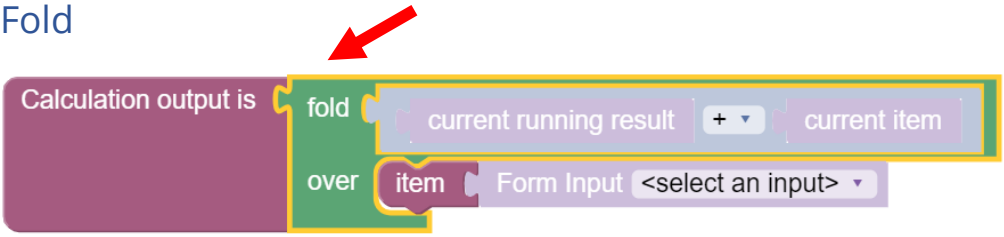
Category: Advanced
What it does: Returns a list of items that have been mapped over provided items.
How to use it: Supply a piece such as the Operator in the "map" parameter to perform an operation on each list item in the "over" parameter. For example, use this piece to add 1 to the current item in the "map" parameter and supply a list from 0 to 10 for the "over" parameter to return a list from 1 to 11, as each item in the initially provided list has had 1 added to it.

Filter



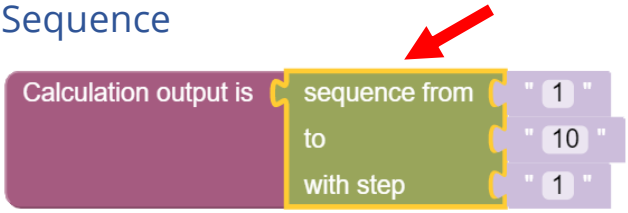
Category: Advanced
What it does: Returns a list of items that have been filtered from a provided list.
How to use it: Supply a Comparer in the "filter by" parameter to return only items that match the criteria provided. For example, use this piece to get a list of even numbers using modulo or filter out undesired values on the fly without having to create a new list elsewhere.

Fold



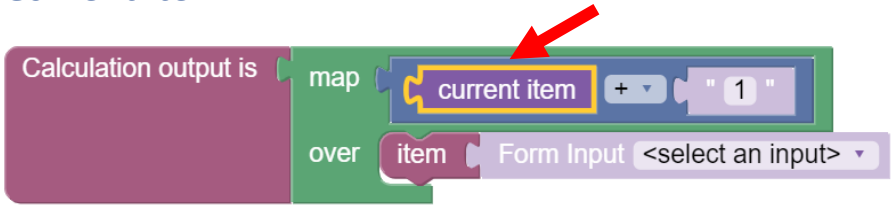
Category: Advanced
What it does: Returns a single value from combining the provided items.
How to use it: Supply a piece such as the Operator in the "fold" parameter to perform an operation on each list item provided in the "over" parameter. For example, use this piece to add the current item to current running result in the "fold" parameter and provide a list from 1 to 5 to return the result of 15 (1 + 2 + 3 + 4 + 5 = 15).

Sequence



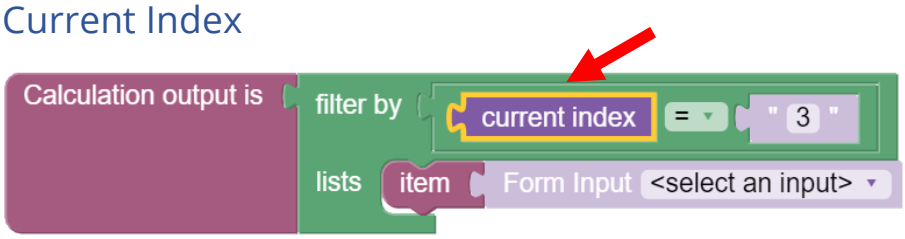
Category: Advanced
What it does: Returns a list of numbers spanning the provided range.
How to use it: Use the "Sequence" piece to generate a range of numbers. Provide the first number of the range in the "sequence from" parameter and the last number of the range in the "to" parameter. The sequence steps by 1 by default, but if every other number is desired for instance, enter 2 into the "with step" parameter.

Current Item



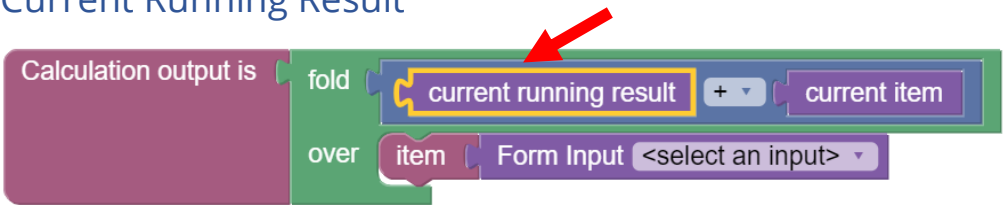
Category: Advanced
What it does: Returns the current item's value when iterating through advanced functions.
How to use it: Supply the "Current Item" piece in an Operator or Comparer in a Map, Filter, or Fold function's first parameter. Each iteration takes the current item's value and performs the supplied operation or comparison to return a new value.

Current Index



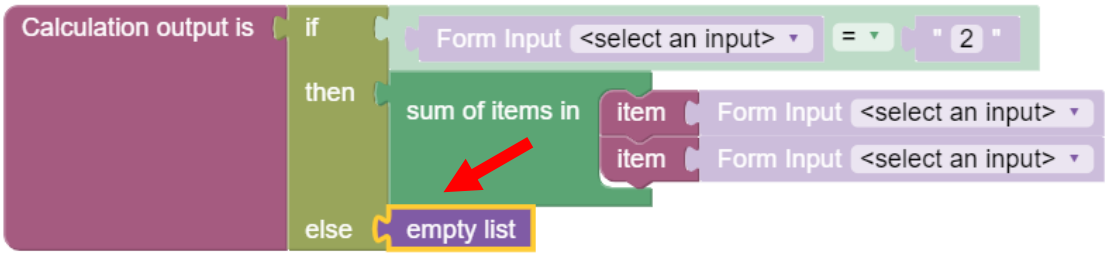
Category: Advanced
What it does: Returns the index of the current item in advanced functions.
How to use it: Use the "Current Index" piece to return the index of the current item in a Map or Filter function. Indexing starts at 1, not 0. For example, use this piece to return a specific item in a list based on index.

Current Running Result



Category: Advanced
What it does: Returns the current running total in each iteration of a Fold function.
How to use it: Supply the "Current Running Result" piece within an Operator for the "fold" parameter. Each iteration takes the current running total and performs the supplied operation, then moves onto the next iteration.

Empty List



Category: Advanced
What it does: Returns an empty list.
How to use it: Use the "Empty List" piece to return an empty list if a condition is not met.

Examples and Tips

This section assumes you have access to Power Editor mode, which is required to add calculations to a form.

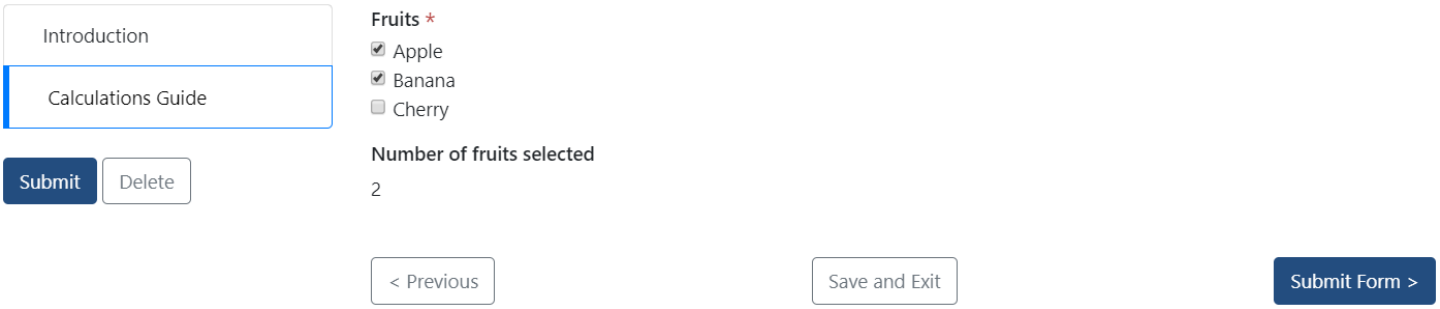
Count the number of options selected in a check group

In this example, the number of options selected by a user is displayed to demonstrate how to count all selected options.

1. Add a check group to the form.
    a. Give it a "Label" of "Fruits."
    b. Supply the following options:
        i. Apple
        ii. Banana
        iii. Cherry
2. Add a calculation to the form.
    a. Give it a "Label" of "Number of fruits selected" and set the "Layout" to "Default" or leave unselected so it displays to the end user.
    a. Return the Count of the "Fruits" Form Input to display the number of selected options.

Page Builder

Web Form



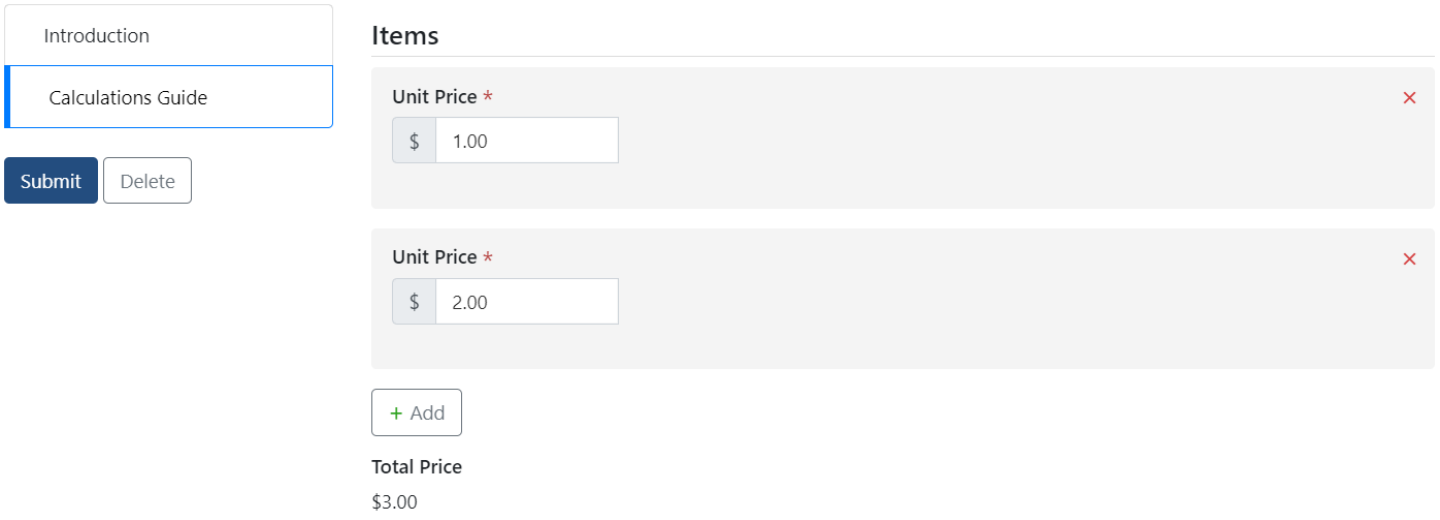
Return the total sum of a value in a repeatable region

In this example, there is a repeatable region that includes a simple Unit Price text box. The calculation returns the sum of all Unit Price text boxes added because of the addition of multiple repeatable regions.

1. Add a repeatable region to the form.
    a. Give it a “Label” of “Items.”
    b. Within the repeatable region, add a text box with a “Label” of “Unit Price” and a “Data Type” of “Currency.”
2. Add a calculation to the form.
    a. Give it a “Label” of “Total Price” and set the “Layout” to “Default” or leave unselected so it displays to the end user.
    b. Use a Concatenation to join a simple Value of “$” to a Sum of the “Unit Price” Form Input with a Decimal Format set to 2 places.

Page Builder

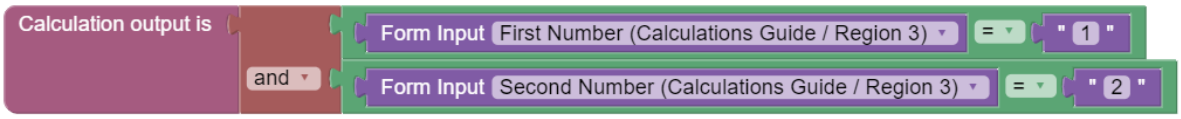
Web Form



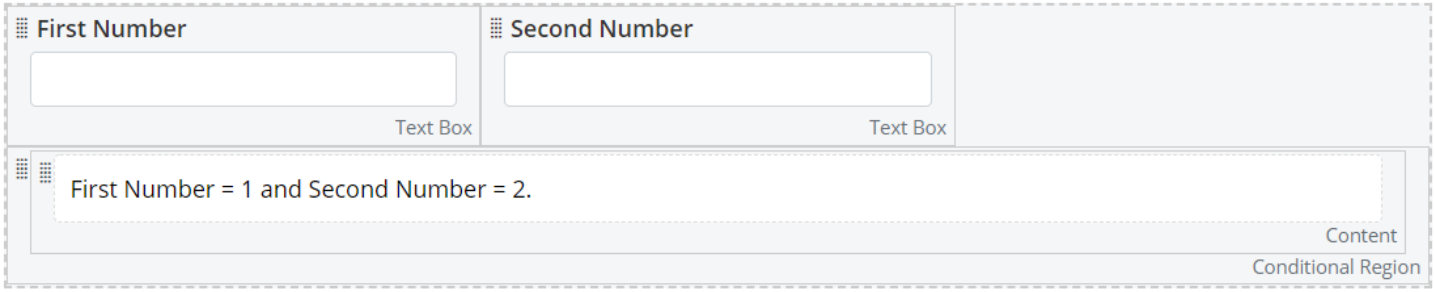
Display a conditional region if multiple conditions must be met

In this example, a content block has been added within the conditional region that displays if both First Number equals 1 and Second Number equals 2. The Calculation property of a conditional region is used to display it.

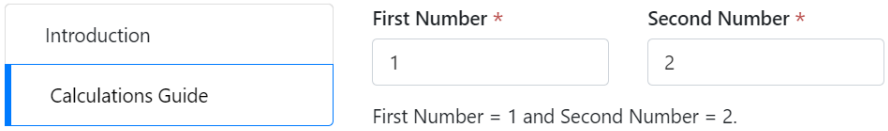
1. Add a conditional region to the form.
    a. Input a calculation under the "DISPLAY IF CALCULATION IS TRUE" section.
    b. Use an IF-ELSE Statement with an AND/OR piece to check if both "First Number" Form Input = 1 AND "Second Number" Form Input = 2. If both are satisfied, then return "true" else return "false."



Page Builder



Web Form



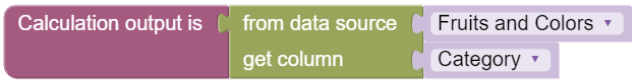
Return a list of distinct values from a data source column

For this example, a column with duplicate values is pulled from a data source and joined with an empty list to return a list with only distinct values.

1. In the "Data Sources" admin tool, create a data source titled "Fruits and Colors."
    a. Create "Category" and "Item" columns with the values shown in the image below.

Category	Item
Fruit	Apple
Fruit	Banana
Fruit	Cherry
Color	Blue
Color	Green
Color	Red

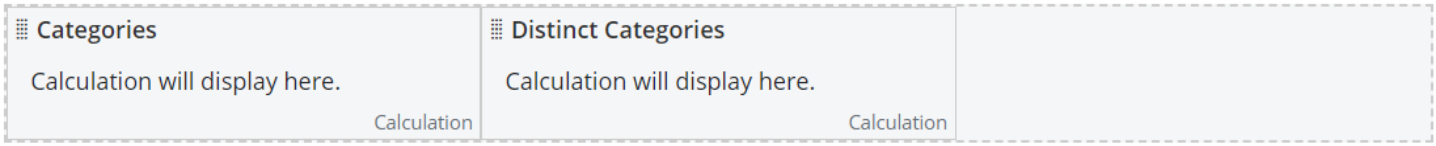
2. Add a calculation to the form.
    a. Give it a "Label" of "Categories" and set the "Layout" to "Default" or leave unselected so it displays to the end user.
    b. Use a From Data Source piece to return the "Category" column from the "Fruits and Colors" data source. This simply returns a list including duplicate values. Continue to step 3.



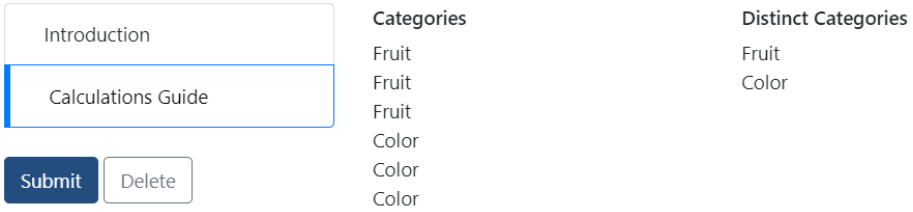
3. Add another calculation to the form.
    c. Give it a "Label" of "Distinct Categories" and set the "Layout" to "Default" or leave unselected so it displays to the end user.
    d. Use a From Data Source piece to return the "Category" column from the "Fruits and Colors" data source. Get only Distinct items of the returned list from the data source. This returns only the unique values from the "Category" column in the data source.

To pass the list of distinct values to a dropdown rather than to a calculation, see Dynamically supply dropdown options based on user input.

Page Builder



Web Form



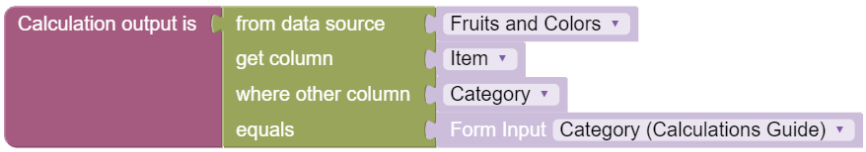
Dynamically supply dropdown options based on user input

For this example, a data source is added with two unique categories and several items associated with each. Selecting a category in a dropdown dynamically generates the list for a second dropdown.

1. In the "Data Sources" admin tool, create a data source titled "Fruits and Colors."
    a. Create "Category" and "Item" columns with the values shown in the image below.

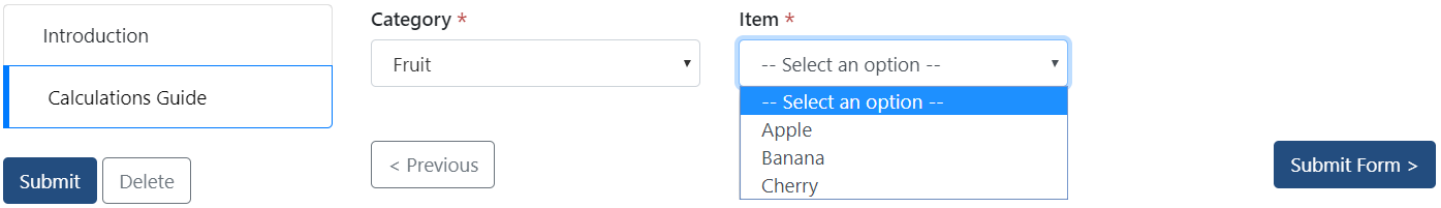
Category	Item
Fruit	Apple
Fruit	Banana
Fruit	Cherry
Color	Blue
Color	Green
Color	Red

2. Add a dropdown to the form.
    a. Give it a "Label" of "Category."
    b. Select "DataSource" for "List Type."
    c. Select "Fruits and Colors" for the "Data Source Table."
    d. Select "Category" for the "Data Source Column."
        i. NOTE: data source columns used in dropdowns only return distinct values.
3. Add another dropdown to the form.
    a. Give it a "Label" of "Item."
    b. Select "Custom" for "List Type."
    c. Add a single custom item under "Items."
    d. Toggle the item to a calculation.
        i. NOTE: toggle the "Label" of the item to a calculation if in Power Editor mode.
    e. "Input a calculation" that uses a From Data Source - Where piece to return the "Item" column from the "Fruits and Colors" data source where the "Category" column equals the "Category" Form Input.



Page Builder

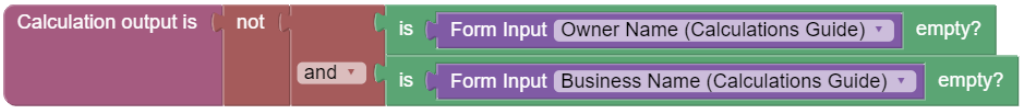
Web Form



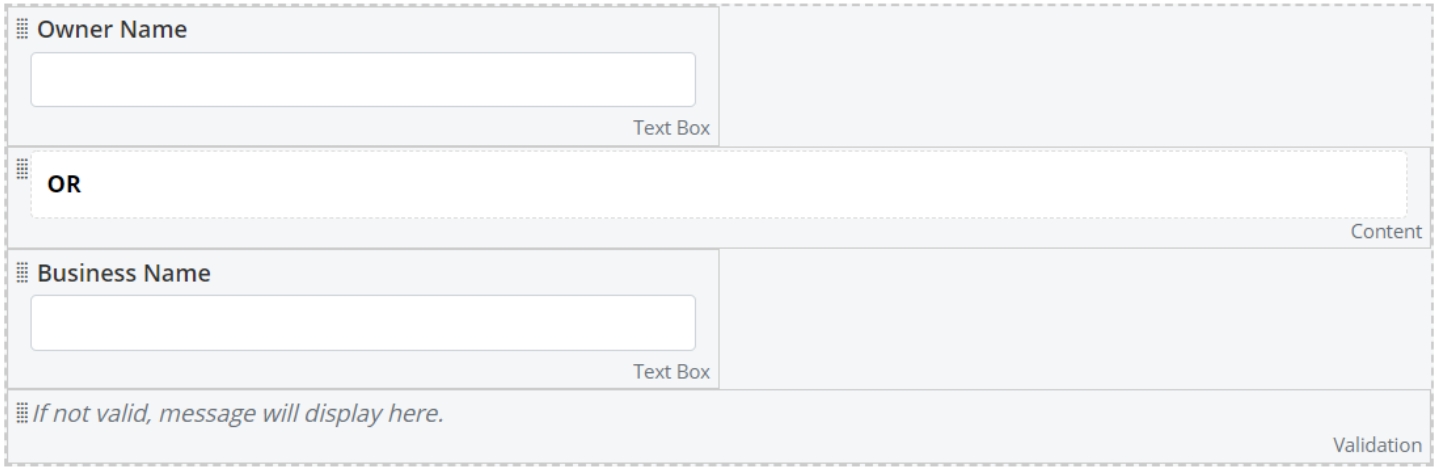
Require user to enter one value or another

For this example, a calculation is added to check if one input or the other has entered data, and additional settings are added to text boxes to display an error message if not satisfied.

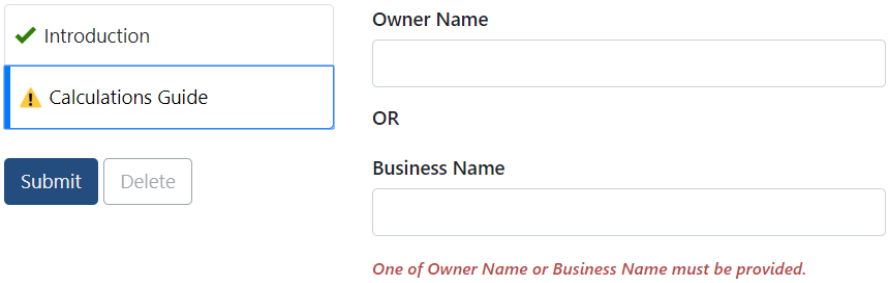
1. Add a text box to the form.
    a. Give it a "Label" of "Owner Name."
    b. Set "Required" to "False."
2. Add a content block to the form.
    a. Type in the word "OR."
3. Add another text box to the form.
    a. Give it a "Label" of "Business Name."
    b. Set "Required" to "False."
4. Add a validation control to the form.
    a. Give it an "Invalid Message" of "One of Owner Name or Business Name must be provided."
    b. In the "Require to be True" calculation, use the NOT, AND/OR, and Is Empty pieces to check if both "Owner Name" and "Business Name" are empty.



Page Builder



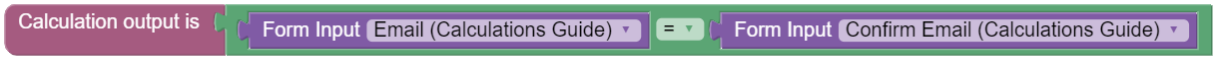
Web Form



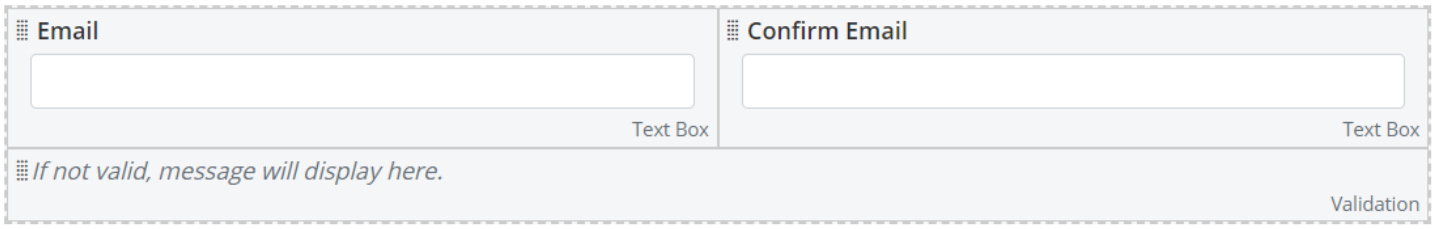
Validate that one text box equals another text box

For this example, custom validation is added to check if one text box equals another text box. A common application of this example is to confirm an email address.

1. Add a text box to the form.
    a. Give it a “Label” of “Email.”
    b. Set “Required” to “True.”
2. Add another text box to the form.
    a. Give it a “Label” of “Confirm Email.”
    b. Set “Required” to “True.”
3. Add a validation control to the form.
    a. Give it an “Invalid Message” of “Emails must match.”
    b. In the “Require to be True” calculation, use a Comparer to require that “Email” = “Confirm Email.”



Page Builder



Web Form