



# Calculations

## Help Guide

Published: 6 March 2025

## Table of Contents

|     |   |    |
|-----|---|----|
| 1.  | Using Calculation.....                                  | 2  |
| 1.1 | Add Calculations to a Form.....                         | 2  |
| 1.2 | Add a Second Calculation to a Date Field.....           | 4  |
| 2.  | Test the Form.....                                      | 8  |
| 2.1 | Open the Submission .....                               | 9  |
| 3.  | Use Address Calculations.....                           | 10 |
| 3.1 | Calculate the City.....                                 | 11 |
| 3.2 | Calculate Eligibility based on State of Residence ..... | 12 |
| 3.3 | Conditional Address Calculation.....                    | 14 |
| 4.  | Counting.....   | 15 |
| 5.  | Date Calculations.....                                  | 17 |
| 5.1 | Find the name of the day of the week.....               | 17 |
| 5.2 | Adding Days.....  | 20 |
| 6.  | Number Calculations.....                                | 21 |
| 6.1 | Fill in the Form.....                                   | 24 |
| 7.  | String Calculations.....                                | 25 |
| 7.1 | IndexOf.....  | 25 |
| 7.2 | Includes.....   | 26 |
| 7.3 | CharCodeAt.....   | 26 |
| 7.4 | SubStr.....   | 27 |
| 7.5 | Left/Right.....   | 28 |
| 7.6 | Length.....   | 28 |
| 7.7 | Concatenation.....                                      | 28 |

# 1. Using Calculation

Calculations can be created in forms to produce variations of data and/or calculate an alternative outcome based on user inputs. Calculations allow you to apply a formula using one or more fields to return new values based on the data entered. This can be used for simple to complex maths calculations, identifying the difference in days or times, calculating dates, among other uses. Calculations can also be used to create text responses based on conditions.

For example, if the form filler has entered their date of birth, a simple calculation can tell us what their age is on a certain date.

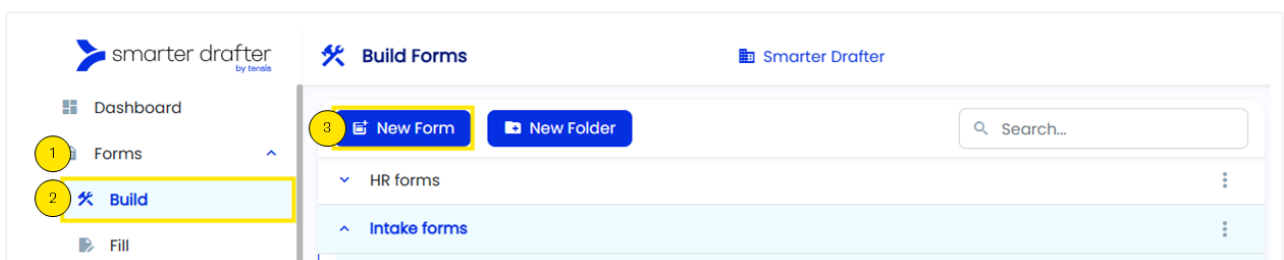
Calculations can be used to manipulate any field type and return alternate data. For example, you can split out date components to tell you the day of week, month or year. You can also calculate the time between dates, and return a measurement in days, weeks, months or years.

Let's look at some examples of calculations.

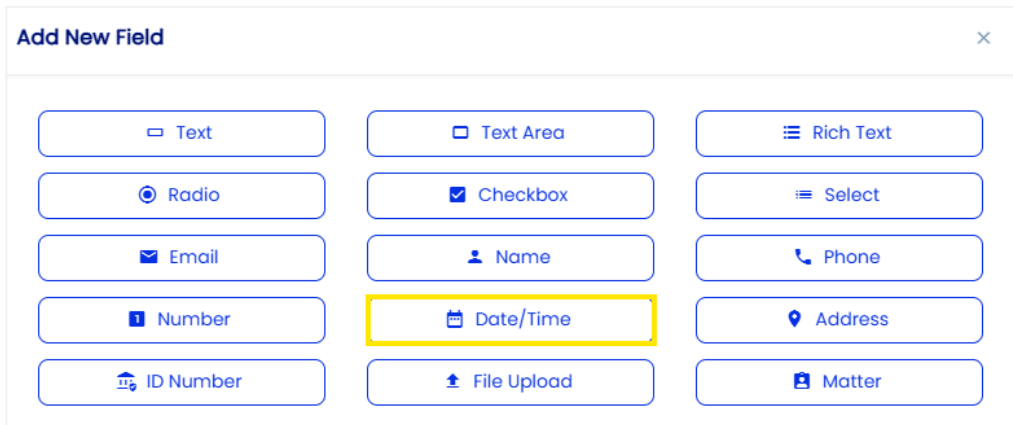
## 1.1 Add Calculations to a Form

In this example, a form will be created which contains a date of birth field for a child. This will be used to calculate a new date. Start by creating a form.

1. Click **Forms**.
2. Click **Build**.
3. Click **New Form**.



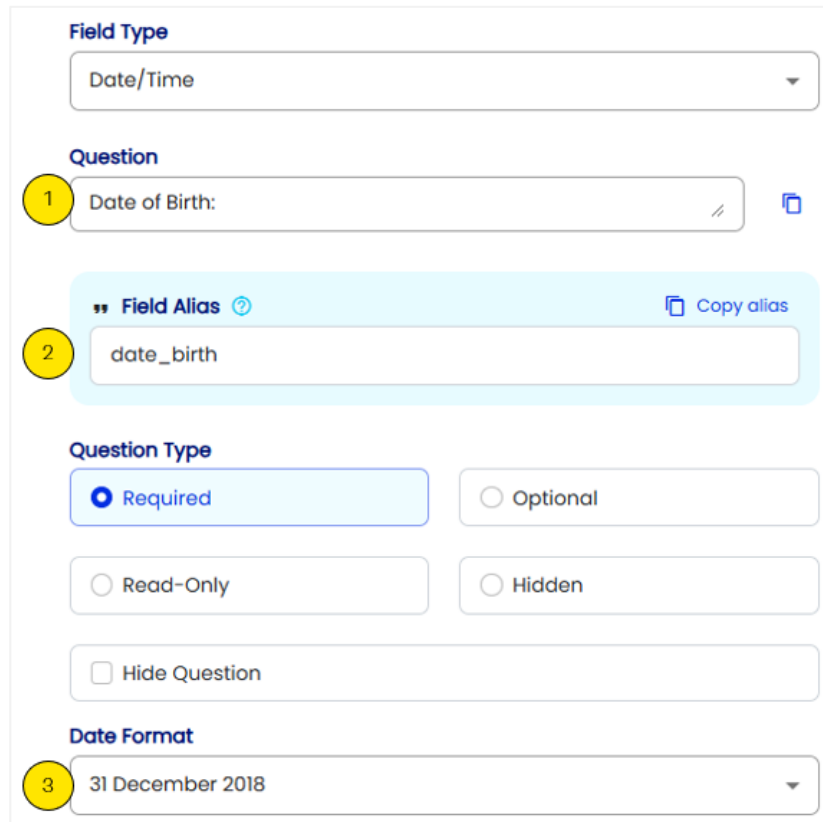
4. Add a Date/Time field.



**Add New Field** [Close]

- Text
- Text Area
- Rich Text
- Radio
- Checkbox
- Select
- Email
- Name
- Phone
- Number
- Date/Time**
- Address
- ID Number
- File Upload
- Matter

1. Add a Question
2. Add a Field Alias.
3. Scroll down and select a Date Format to display. In this example, the format is: 31 December 2018.



**Field Type**  
Date/Time

**Question**  
1 Date of Birth:

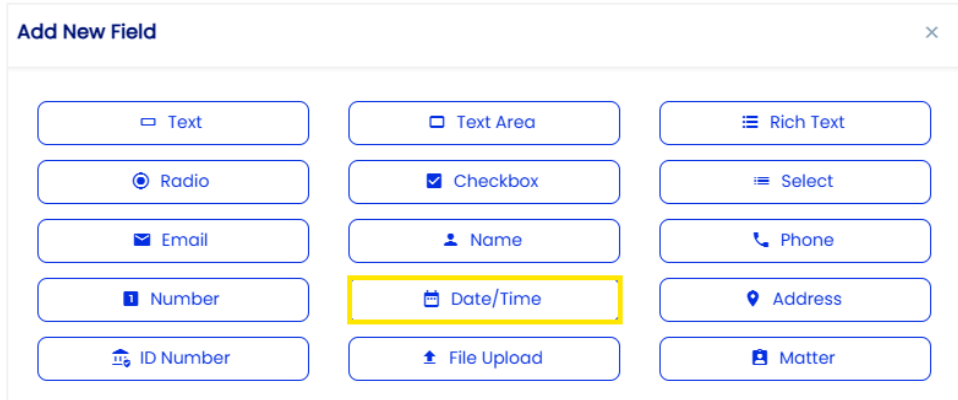
**Field Alias** [Copy alias](#)  
2 date\_birth

**Question Type**  
 Required  Optional  
 Read-Only  Hidden  
 Hide Question

**Date Format**  
3 31 December 2018

## 1.2 Add a Second Calculation to a Date Field

To create a calculation, two fields must interact. Another **Date/Time** field will be added to the form underneath the **Date of Birth** field.

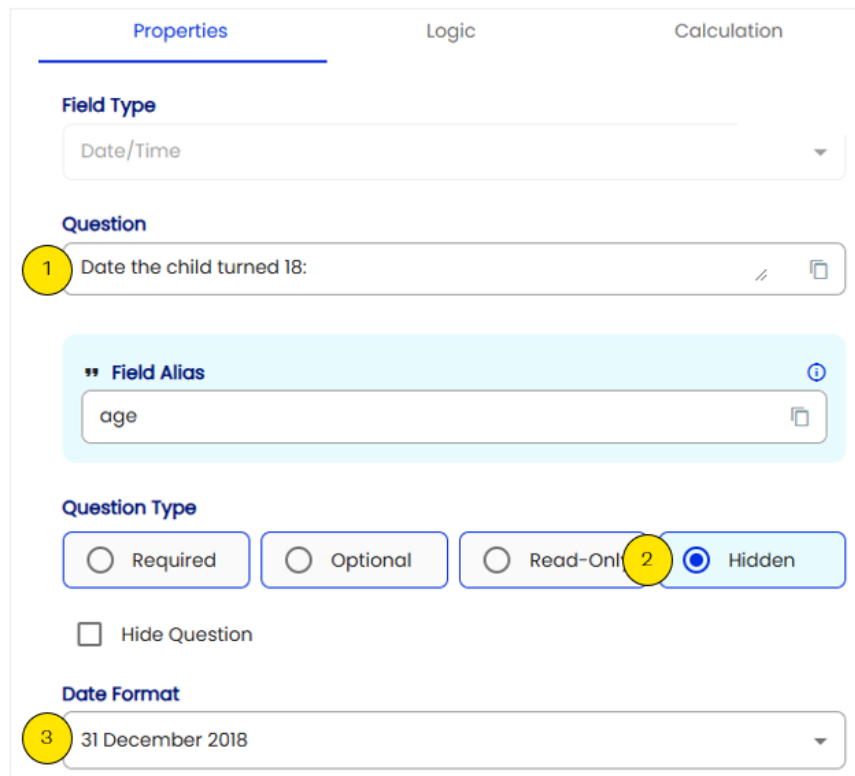


The 'Add New Field' dialog box displays a grid of field types. The 'Date/Time' field type is highlighted with a yellow border.

|  |   |                                    |
|--|---|------------------------------------|
| <input type="checkbox"/> Text          | <input type="checkbox"/> Text Area            | <input type="checkbox"/> Rich Text |
| <input checked="" type="radio"/> Radio | <input checked="" type="checkbox"/> Checkbox  | <input type="checkbox"/> Select    |
| <input type="checkbox"/> Email         | <input type="checkbox"/> Name                 | <input type="checkbox"/> Phone     |
| <input type="checkbox"/> Number        | <input checked="" type="checkbox"/> Date/Time | <input type="checkbox"/> Address   |
| <input type="checkbox"/> ID Number     | <input type="checkbox"/> File Upload          | <input type="checkbox"/> Matter    |

Set Properties to this field:

1. Add a Question: Date the child turned 18.
2. Make the Question Hidden.
3. Select a Date Format.

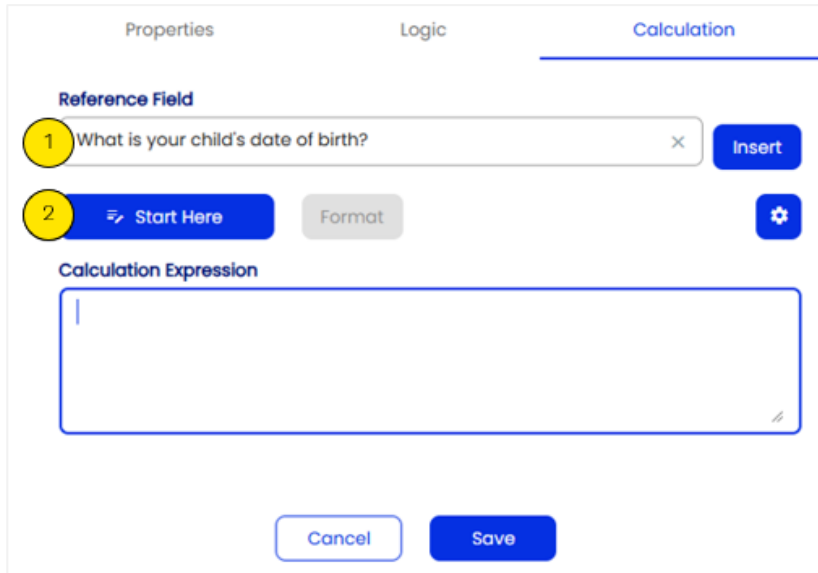


The 'Properties' tab of the field configuration shows the following settings:

- Field Type:** Date/Time
- Question:** Date the child turned 18: (marked with a yellow circle 1)
- Field Alias:** age
- Question Type:** Hidden (radio button selected, marked with a yellow circle 2)
- Date Format:** 31 December 2018 (marked with a yellow circle 3)

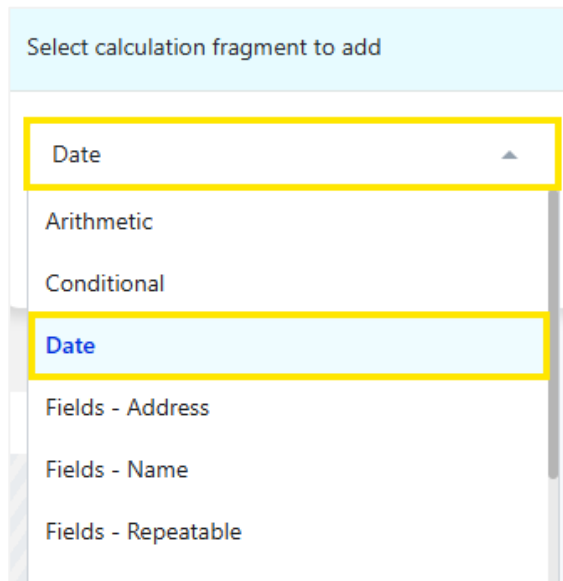
Open the **Calculation** tab. The calculation that will be added here will find the date that the child turned 18.

1. Select the **Reference Field**. (This is the field that the calculation will be based on.)
2. Click **Start Here**.



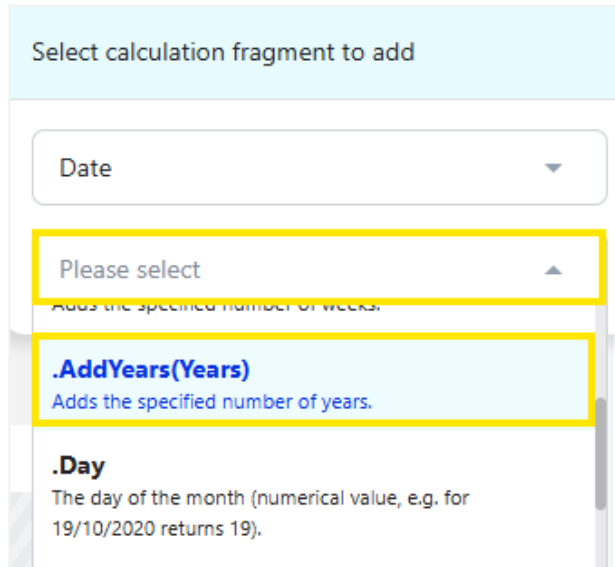
The screenshot shows the 'Calculation' tab in the software interface. At the top, there are three tabs: 'Properties', 'Logic', and 'Calculation'. The 'Calculation' tab is active. Below the tabs, there is a section titled 'Reference Field' with a text input field containing 'What is your child's date of birth?'. A yellow circle with the number '1' is next to the input field. To the right of the input field is an 'Insert' button. Below the input field, there is a 'Start Here' button with a right-pointing arrow, a 'Format' button, and a settings gear icon. A yellow circle with the number '2' is next to the 'Start Here' button. Below these buttons is a large empty text area labeled 'Calculation Expression'. At the bottom of the interface, there are 'Cancel' and 'Save' buttons.

3. A pop up opens. There are two drop-down lists – these are used to build the calculation. Select **Date** in the top menu, as we are calculating a date in the future.

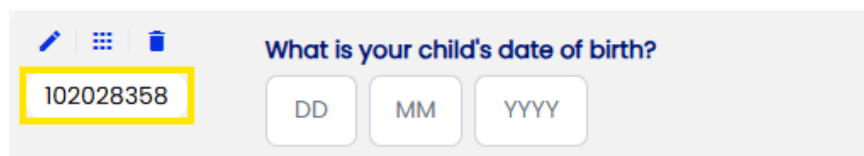


The screenshot shows a dialog box titled 'Select calculation fragment to add'. It contains a list of options: 'Date', 'Arithmetic', 'Conditional', 'Date', 'Fields - Address', 'Fields - Name', and 'Fields - Repeatabe'. The 'Date' option at the top is highlighted with a yellow box. The 'Date' option further down is also highlighted with a yellow box and has a blue background, indicating it is selected.

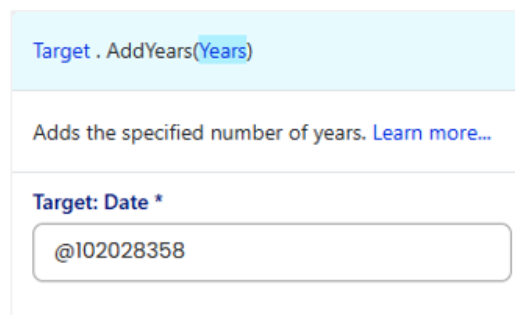
- In the second menu, the command will be selected. For this example, we wish to add 18 years to the date of birth, so the **AddYears** command will be used.



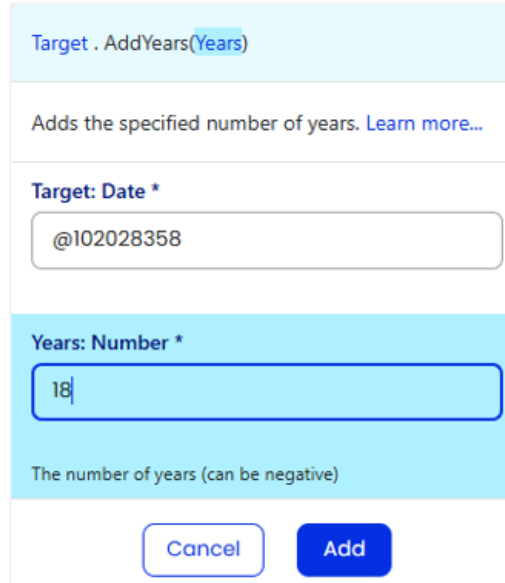
- Now add parameters. In the **Target: Date** field, the field ID number is placed. To find the field ID number, look at the field with the question, "What is your child's date of birth?". The field ID number is in the bottom left corner (look at the example below: the ID is 102028358). Click the number to copy it to the clipboard.



- Paste the field ID number in the **Target:Date** field.



- In the Years: Number field, we must add the number of years to be added to the target date. The number of years to be added is 18.



Target . AddYears(Years)

Adds the specified number of years. [Learn more...](#)

Target: Date \*

@102028358

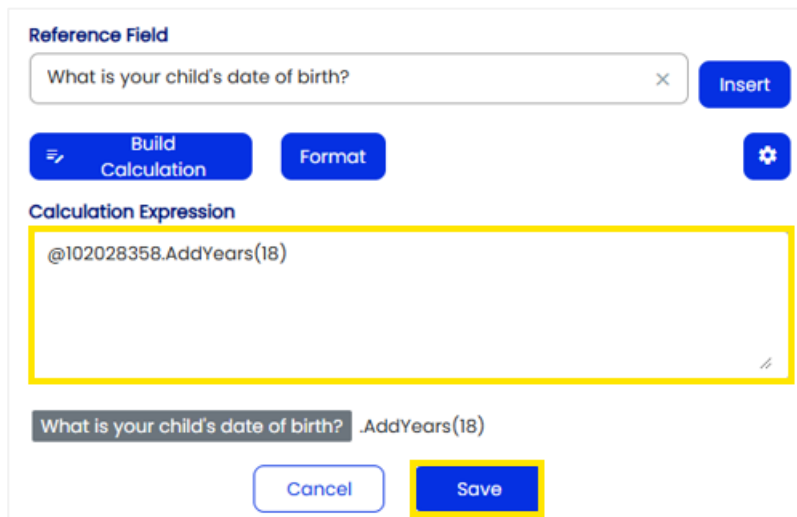
Years: Number \*

18

The number of years (can be negative)

Cancel Add

- The Calculation Expression is created for you by Smarter Drafter and shown in the builder.
- Click Save.



Reference Field

What is your child's date of birth? x Insert

Build Calculation Format

Calculation Expression

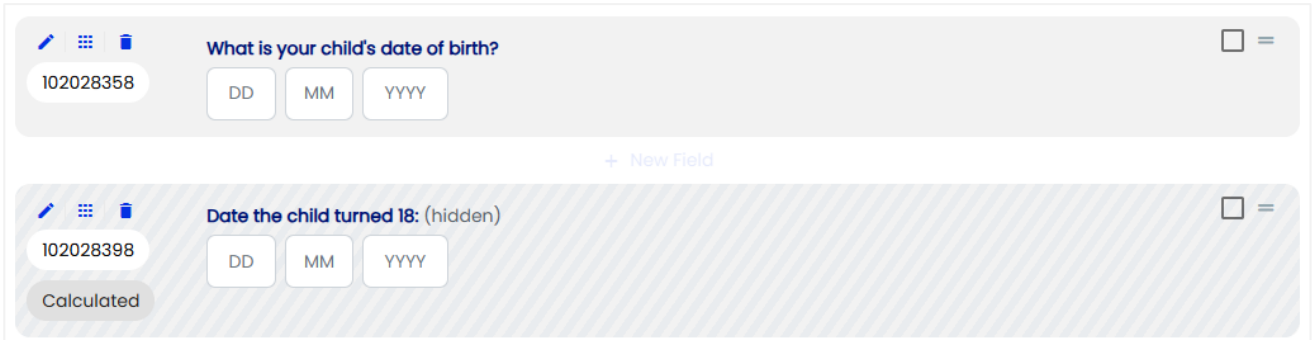
@102028358.AddYears(18)

What is your child's date of birth? .AddYears(18)

Cancel Save



10. The calculation field is now built into the form, underneath the question, "What is your child's date of birth?". The field is hidden, so it will not be visible when the form is filled out as there's no need for the form filler to see this. It will be visible in the submission, after the form is submitted.

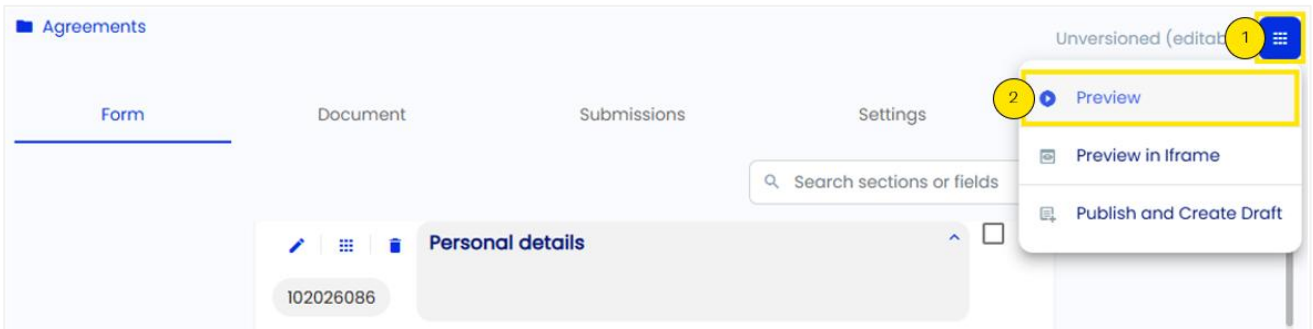


The screenshot shows two form fields in a builder interface. The first field is titled "What is your child's date of birth?" and contains the value "102028358". It has three input boxes labeled "DD", "MM", and "YYYY". Below it is a "+ New Field" button. The second field is titled "Date the child turned 18: (hidden)" and contains the value "102028398". It also has "DD", "MM", and "YYYY" input boxes. A "Calculated" label is visible below the value. Both fields have a waffle menu icon and a trash icon in the top left corner, and a checkbox with an equals sign in the top right corner.

## 2. Test the Form

If we preview this form, we will not see the hidden field. It will be filled out, and the calculation field will deliver data in the submission. Let's see how it works.

1. Click the **waffle** button.
2. Click **Preview**.



The screenshot shows the "Agreements" form builder interface. The "Form" tab is selected. A "Personal details" field is visible with the value "102026086". A dropdown menu is open, showing options: "Preview", "Preview in Iframe", and "Publish and Create Draft". The "Preview" option is highlighted. The interface includes a search bar for "Search sections or fields" and a "Unversioned (edit)" button in the top right corner.

- We can now see the form, ready to be filled out. The age calculation field is set to be hidden, so we do not see it here, but it is embedded underneath the child's date of birth field. Fill in the form and click Submit Form.

### Personal details

Please give us some details about yourself, and we will get in touch.

**Name:**

Use Placeholder

**Date of Birth:**

**What is your child's date of birth?**

**Email:**

Use Placeholder

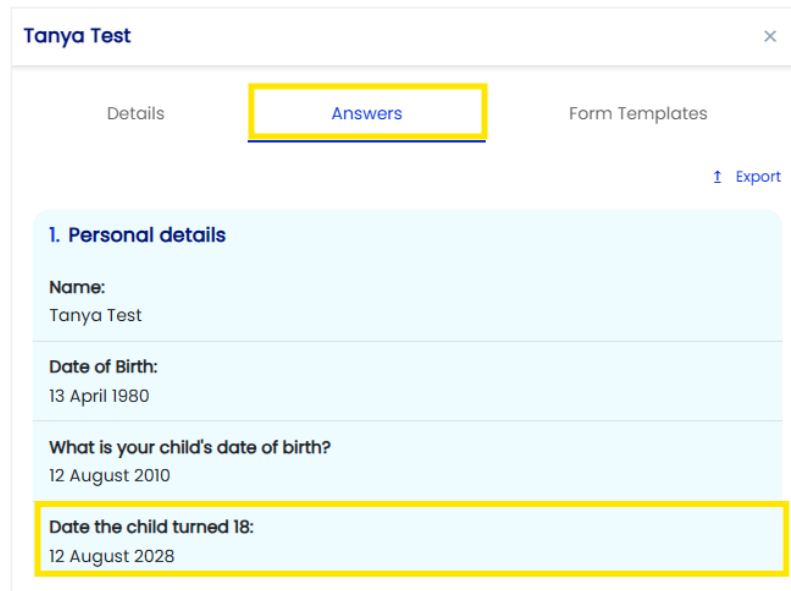
## 2.1 Open the Submission

When the form is filled out and submitted, the data can be viewed in the **Submissions** tab. In this example, the name of the contact in the form is Tanya Test.

- Click the **Submission** tab.
- Click the **Submission Name** (Tanya Test).

| Form                     | Document                     | 1 Submissions          | Settings                               | Role Mapping        |           |                       |         |
|--------------------------|------------------------------|------------------------|--|---------------------|-----------|-----------------------|---------|
| <a href="#">Reload</a>   | <a href="#">E-Signatures</a> | <a href="#">Export</a> | <input type="text" value="Search..."/> |                     |           |                       |         |
| <input type="checkbox"/> | Matter                       | Client(s)              | Submission Name                        | Date ↓              | Status    | Saved or Submitted By | Company |
| <input type="checkbox"/> |                              |                        | 2 Tanya Test                           | 12 Dec 2024 5:57 PM | Generated | Carmen Brooks         |         |

- The submission opens, and the responses can be viewed in the right-hand side of the screen. The **hidden calculation field** can be seen here, where the date the child reaches 18 is calculated (in this example, the child is currently under 18, so the date is in the future).



**Tanya Test**

Details **Answers** Form Templates

↑ Export

**1. Personal details**

**Name:**  
Tanya Test

**Date of Birth:**  
13 April 1980

**What is your child's date of birth?**  
12 August 2010

**Date the child turned 18:**  
12 August 2028

---

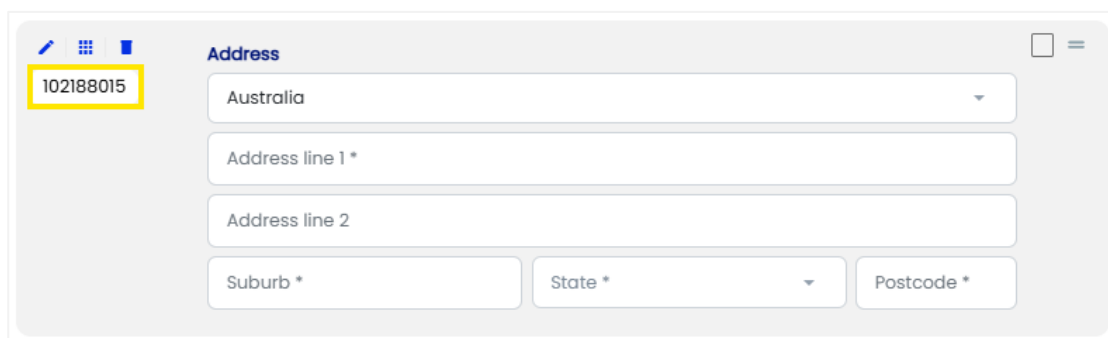
**Note:** A similar calculation can be used to calculate months after a date, using `.AddMonths(Months)`.

---

### 3. Use Address Calculations

Address calculations can be used to extract a part of an address. This is useful to identify if an address falls within a legal jurisdiction, for example.

Let's look at how to use this type of address calculations, with this address field as the example (target field ID: 102188015):



**102188015**

**Address**

Australia

Address line 1 \*

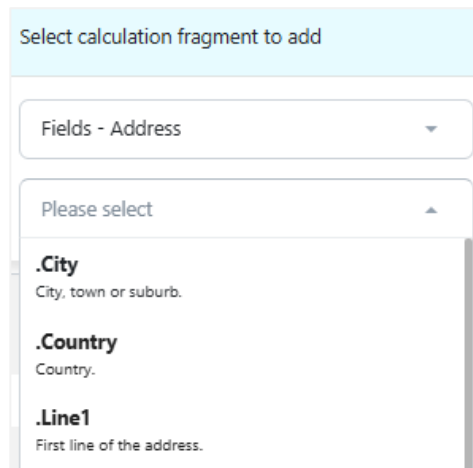
Address line 2

Suburb \* State \* Postcode \*

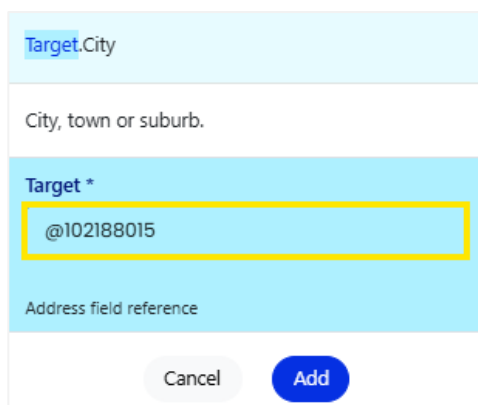
### 3.1 Calculate the City

To extract a city/suburb name by performing an address calculations:

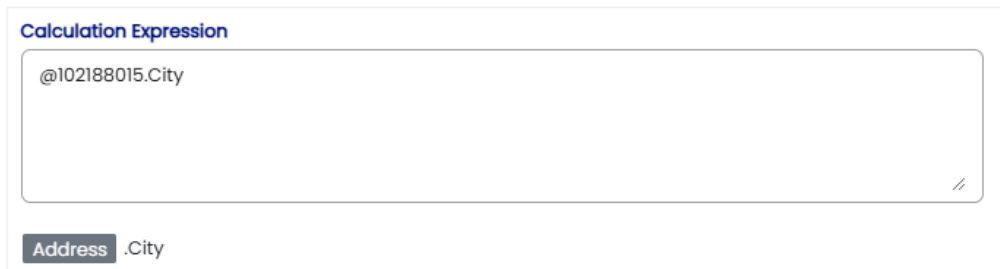
1. Create a **new field** underneath the address field.
2. Select **Text** for the field type and set the field to **read-only**. This will show the field to the form filler, but not allow it to be typed in.
3. Click the **Calculation** tab.
4. Click **Start Here**. Select **Fields – Address** from the first list.
5. Select **.City** from the second list.



6. The **Target field** needs to be populated with the field ID for the target address field (above). Click to copy the field ID (in this example, 102188015).



7. The calculation expression is created in the Calculation Expression field.



8. You can target a different part of the address by using a different fragment of the address. For example, to extract the country, do the same process above, but use .Country. Here's a full list of the fragments available:

- .City
- .Country
- .Line1
- .Line2
- .Postcode
- .State

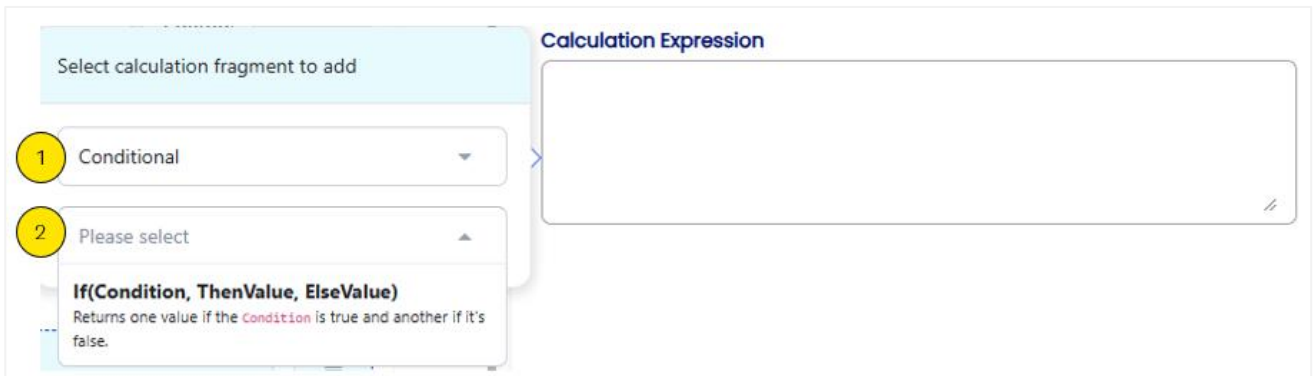
### 3.2 Calculate Eligibility based on State of Residence

In this example, conditional calculation will be applied to a calculated field to inform a form filler whether they are eligible for a program, based on their state. An address field is added, followed by a hidden calculated field to identify the state:



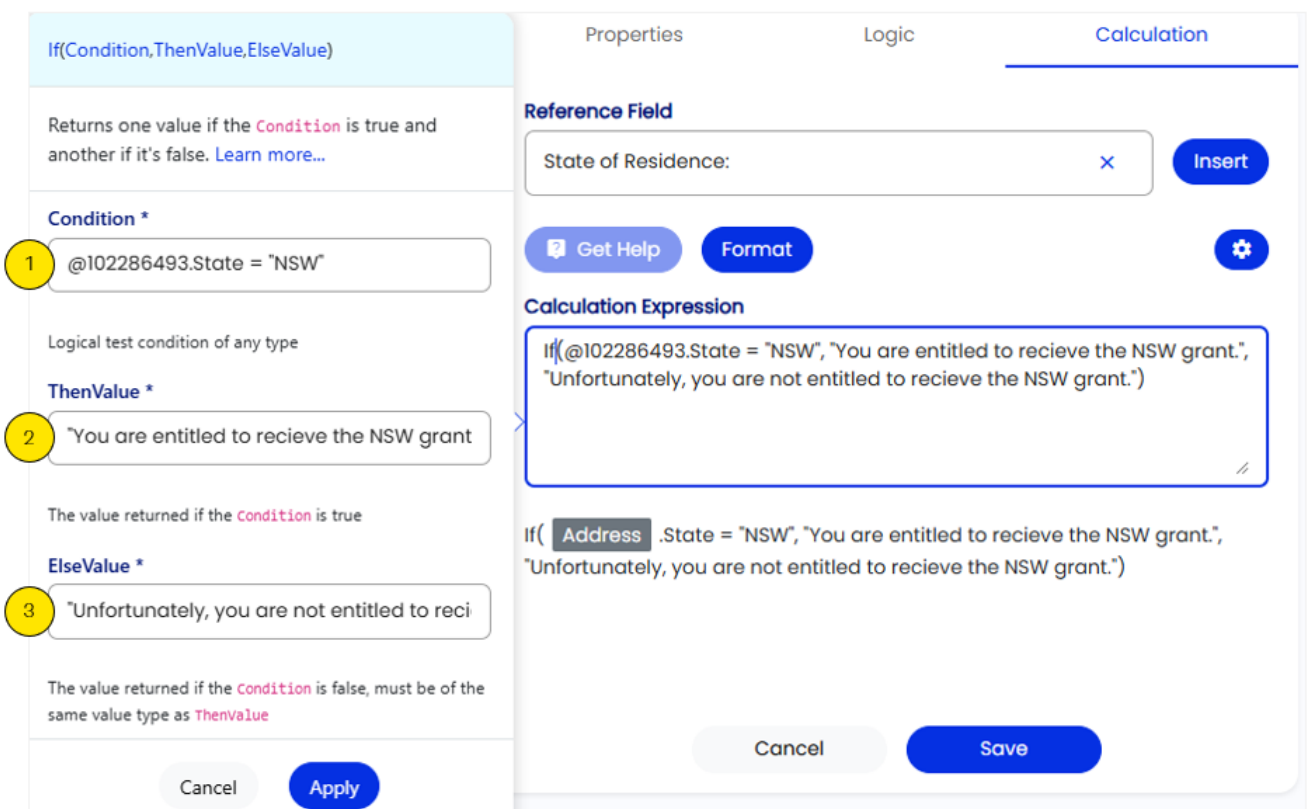
The conditional calculation tab is set up as follows.

1. Start by selecting a conditional type of calculation in the first drop-down list.
2. Select an "If" condition in the second drop-down list.



Add further conditional details.

1. Include the target ID field, the part of address to be addressed, and an operator of "=" and a value of "NSW".
2. In ThenValue, enter the message to be shown if the address is in NSW.
3. In ElseValue, enter the message to be shown if the address is not in NSW. Click Apply.



In the finished form, the calculated field looks like this:

**Address**

Australia

Start typing here ...

123 Pitt Street

Address line 2

Sydney

New South Wales

2000

Use Placeholder

**Eligibility for grant:** *(calculated)*

You are entitled to receive the NSW grant.

### 3.3 Conditional Address Calculation

- In this example, we will create a field that returns the capital city of the state that is selected in the address. To do this, some conditions need to be added in the Calculation Expression, as follows:

```

If(@101392295.State = "ACT", "Canberra",
If(@101392295.State = "NSW", "Sydney",
If(@101392295.State = "NT", "Darwin",
If(@101392295.State = "QLD", "Brisbane",
If(@101392295.State = "SA", "Adelaide",
If(@101392295.State = "TAS", "Hobart",
If(@101392295.State = "VIC", "Melbourne",
If(@101392295.State = "WA", "Perth", ""))))))
    
```

**Calculation Expression**

```

If(@101392295.State = "ACT", "Canberra",
If(@101392295.State = "NSW", "Sydney",
If(@101392295.State = "NT", "Darwin",
If(@101392295.State = "QLD", "Brisbane",
If(@101392295.State = "SA", "Adelaide",
    
```

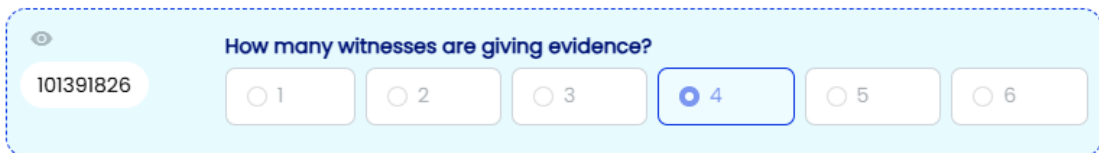
In the finished form, the read-only field will display the capital city of the selected state.

## 4. Counting

If data has been captured via a radio button set or other type of option set, that data can be used to perform calculations. There are four calculation types:

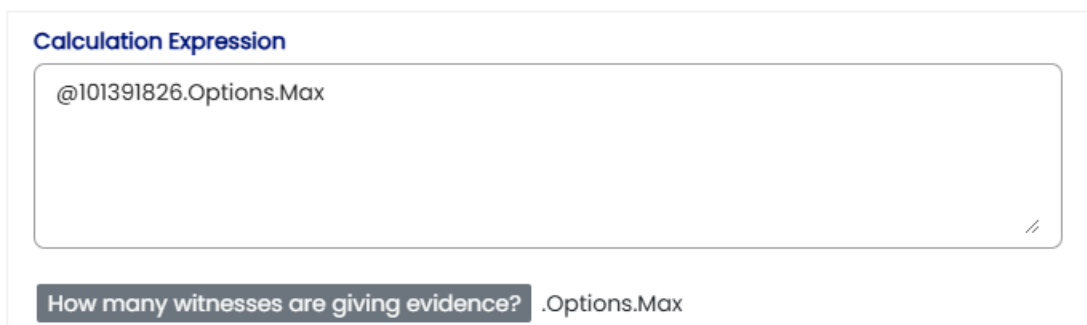
- Count**  
 Count the number of options available in a radio/select/checkbox list and/or Count the number of options selected in a checkbox multi select list
- Max**  
 Display the option with the maximum value in a radio/select/checkbox list and/or Return the value of the maximum option selected in a checkbox multi select list
- Min**  
 Display the option with the minimum value in a radio/select/checkbox list and/or Return the value of the minimum option selected in a checkbox multi select list
- Sum**  
 Calculate the sum of all options in a radio/select/checkbox list and/or Calculate the sum of the options selected in a checkbox multi select list

Here's an example:



The data above can be used in several types of calculations.

- The **.Max** calculation will return the maximum number in the above option set. To use the calculation, target the field ID (101391826), and add the expression, as shown:





This calculation expression will return the maximum number offered by the option set – in this example, 6.

*Result for the radio button - Number of witnesses: (calculated)*

6

- Here's an option set of check boxes against the days of the week.

101391828

**What days are all witnesses available?**

Monday

Tuesday

Wednesday

Thursday

Friday

This data can be used in a calculation to return the maximum possible days available:

**Calculation Expression**

@101391828.Options.Max

What days are all witnesses available? .Options.Max

This field will present in the filled form as follows:

*Result for checkbox - Days available: (calculated)*

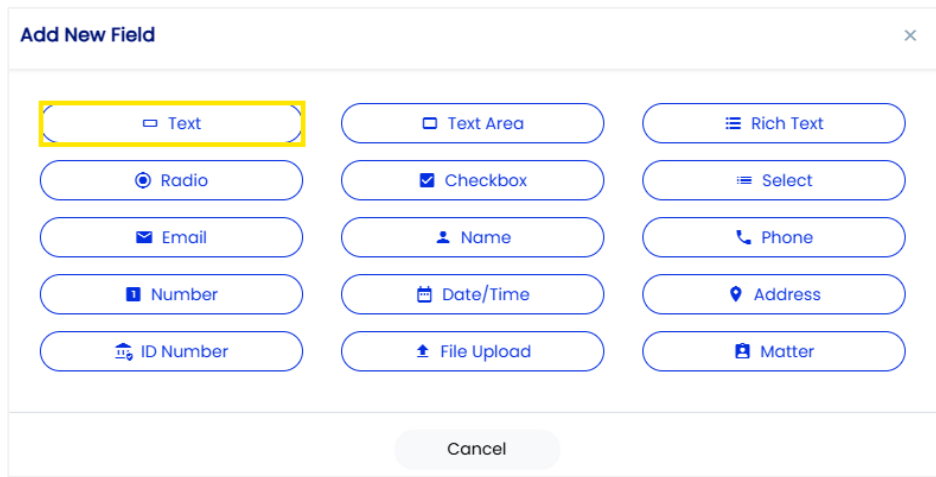
5

## 5. Date Calculations

### 5.1 Find the name of the day of the week

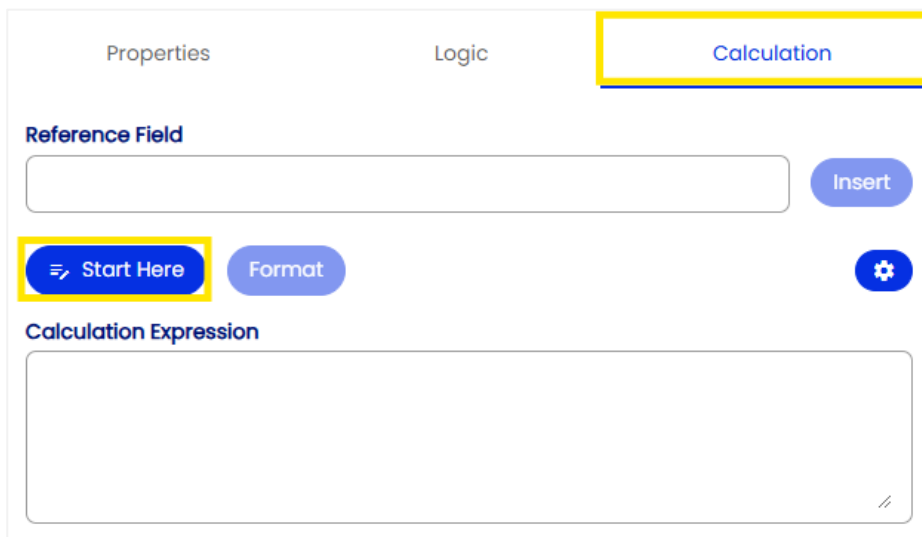
This calculation will show the day of the week of a certain date.

1. Create a new **text field**.



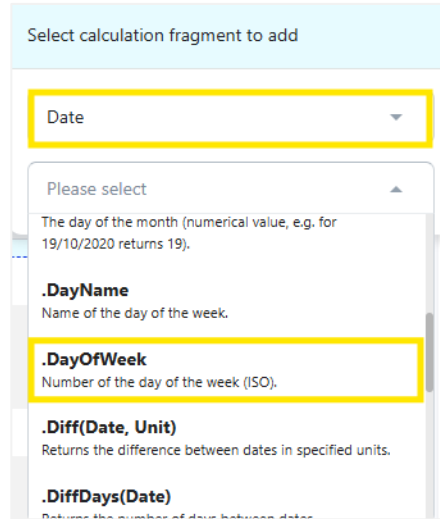
The screenshot shows a dialog box titled "Add New Field" with a close button (X) in the top right corner. It contains a grid of 18 field type options, each with an icon and a label. The "Text" option is highlighted with a yellow border. The options are: Text, Text Area, Rich Text, Radio, Checkbox, Select, Email, Name, Phone, Number, Date/Time, Address, ID Number, File Upload, and Matter. A "Cancel" button is located at the bottom center of the dialog.

2. In the new field editor, click **Calculation** – this will open the calculation builder.
3. Click **Start Here** to begin building the calculation.

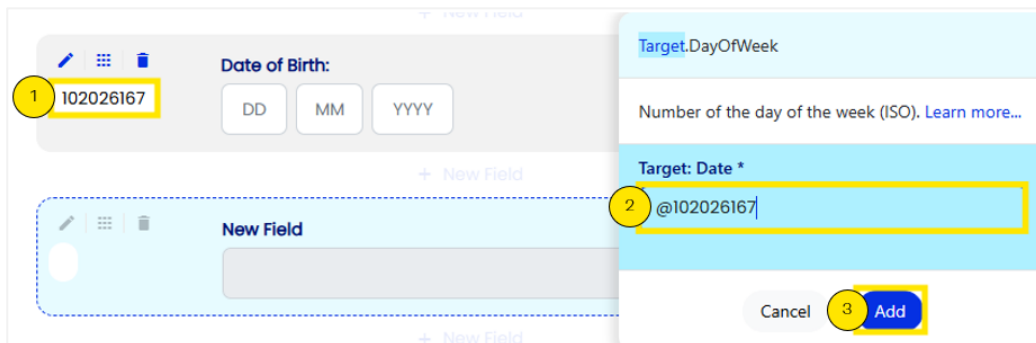


The screenshot shows the field editor interface with three tabs: "Properties", "Logic", and "Calculation". The "Calculation" tab is highlighted with a yellow border. Below the tabs, there is a "Reference Field" section with an empty input box and an "Insert" button. Below that is a "Start Here" button, which is highlighted with a yellow border, and a "Format" button. To the right of the "Start Here" button is a gear icon. Below these buttons is a "Calculation Expression" section with a large empty text area for entering the calculation formula.

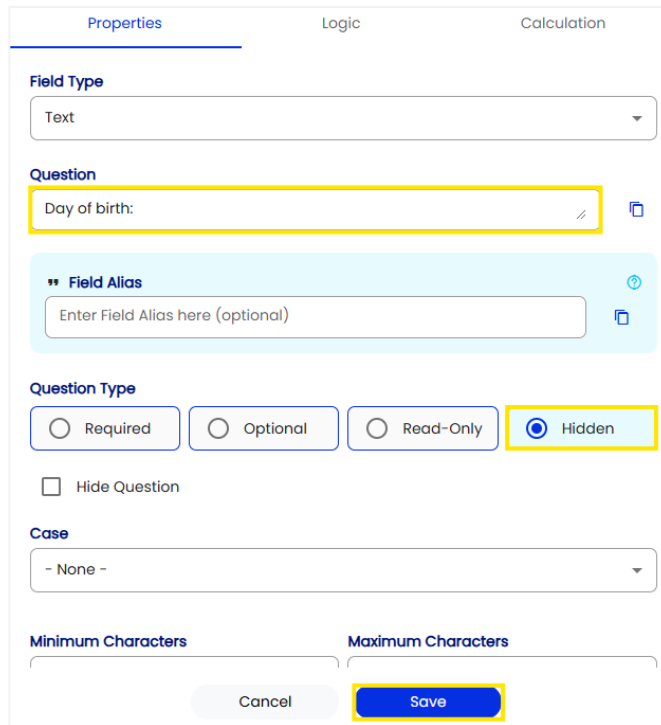
1. Select **Date**.
2. Select **DayofWeek**. This will return the number of the day of the week, starting with Monday at 1.



4. In the **Target: Date**, you may click a **field reference number** to copy it, and
5. Paste it into the **Target: Date** field. You may also type "Today" in the **Target: Date** field, which will find the day of the week for today.
6. Click **Add**.

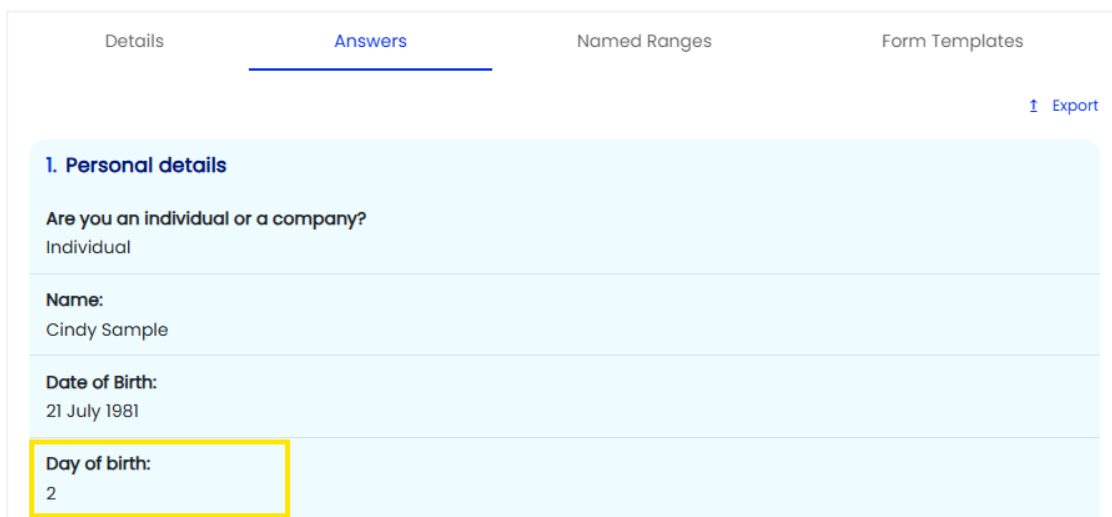


7. In the Properties tab, give the field a Question (to identify the contents of the field).
8. Make a calculation field Read-Only or Hidden.
9. Click Save.



The screenshot shows the 'Properties' tab of a field configuration interface. The 'Field Type' is set to 'Text'. The 'Question' field contains 'Day of birth:'. The 'Field Alias' field is empty. Under 'Question Type', the 'Hidden' radio button is selected. The 'Case' dropdown is set to '- None -'. At the bottom, there are 'Minimum Characters' and 'Maximum Characters' input fields, and 'Cancel' and 'Save' buttons. The 'Save' button is highlighted with a yellow border.

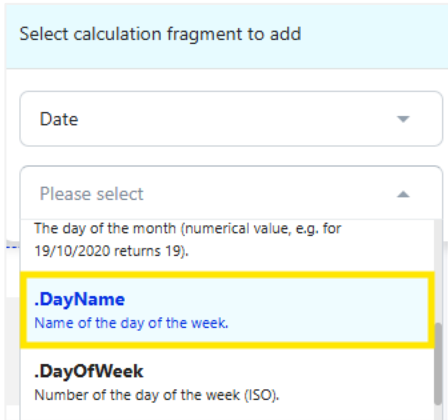
Results: If the form is filled by someone with a date of birth of 21<sup>st</sup> July 1981, the **Answers** screen on the **Submission** would show the Day of birth is 2 (which represents Tuesday).



The screenshot shows the 'Answers' screen of a submission. It features tabs for 'Details', 'Answers', 'Named Ranges', and 'Form Templates'. An 'Export' link is visible in the top right. The content is organized into sections: '1. Personal details', 'Are you an individual or a company?' (answered 'Individual'), 'Name:' (answered 'Cindy Sample'), 'Date of Birth:' (answered '21 July 1981'), and 'Day of birth:' (answered '2'). The 'Day of birth:' field and its answer are highlighted with a yellow border.

Alternatively, if we were to use the `.DayName` calculation in a similar way, the name of the day of a given date would be returned.

The calculation:

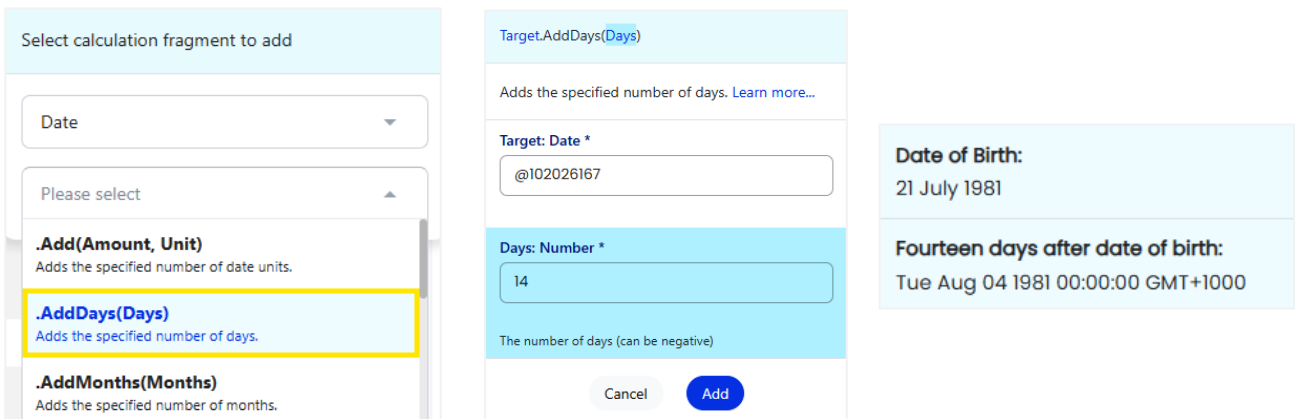


The result in the Submission:



## 5.2 Adding Days

We can calculate what date would occur a certain number of days from a given date by using the `.AddDays(Days)` calculation. In this example, we will add 14 days to the date entered in the `Date of birth` field (identified by its field ID number). A date 14 days after the date of birth is returned in the `Submission`:

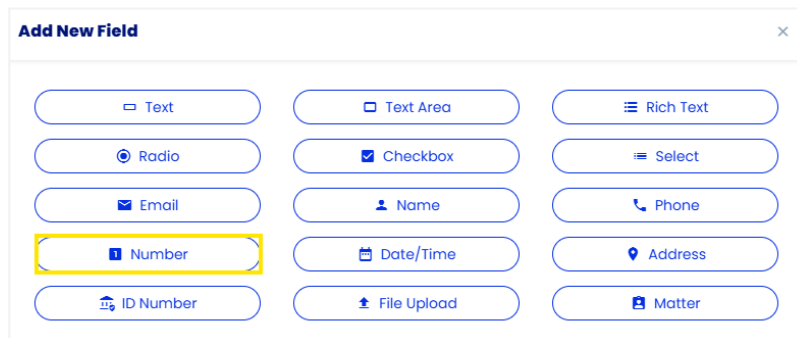


Note: To perform a subtraction, follow the steps above and put a minus sign in front of the number. For example '-14' will return a date two weeks in the past.

## 6. Number Calculations

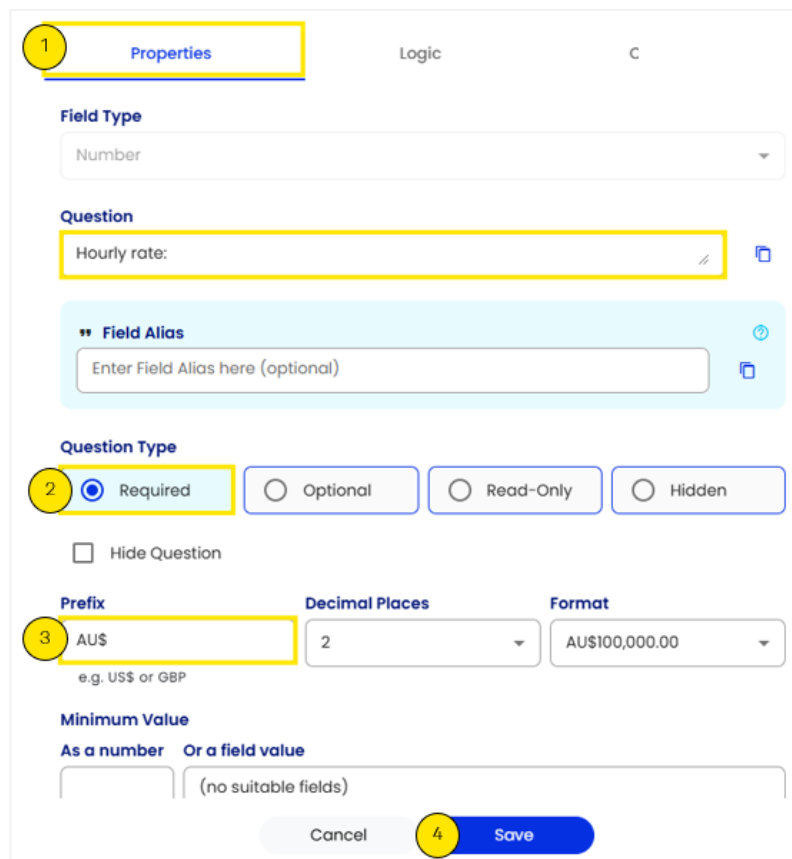
Smarter Drafter can complete arithmetic within a form. Here's a simple example that calculates an employee's pay based on the hourly rate multiplied by the hours worked.

First, create a field to capture the hourly rate. Create a **Number** field.



The 'Add New Field' dialog box displays a grid of field type options. The 'Number' option is highlighted with a yellow border. Other options include Text, Text Area, Rich Text, Radio, Checkbox, Select, Email, Name, Phone, Date/Time, Address, ID Number, File Upload, and Matter.

1. Open the Properties tab.
2. Add a Question to prompt the form filler to enter the hourly rate.
3. Since this is a pay calculation, select a currency as the Prefix.
4. Click Save.



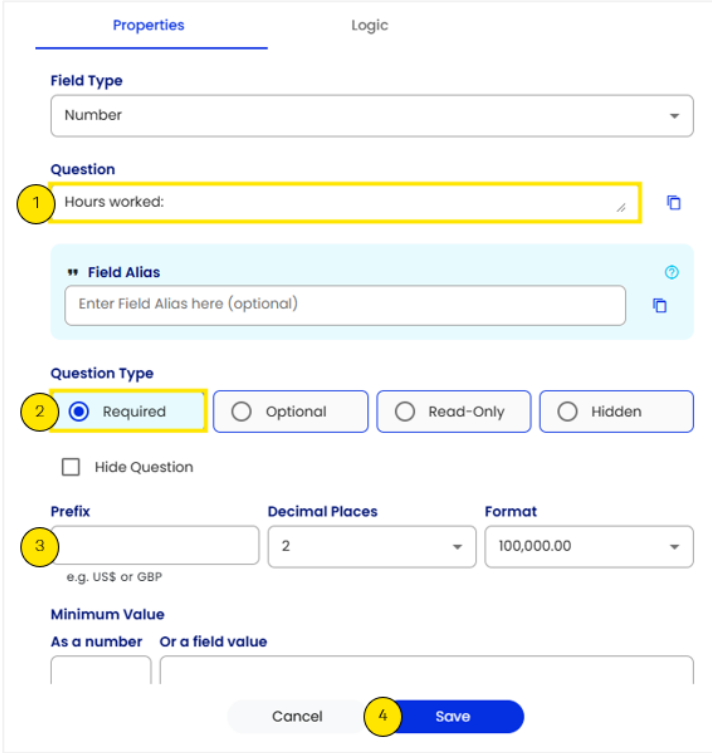
The configuration interface shows the following settings:

- 1**: The 'Properties' tab is selected and highlighted with a yellow box.
- 2**: The 'Field Type' is set to 'Number'.
- 3**: The 'Question' field contains the text 'Hourly rate:' and is highlighted with a yellow box.
- 4**: The 'Question Type' is set to 'Required'.
- 5**: The 'Prefix' is set to 'AU\$'.
- 6**: The 'Decimal Places' is set to '2'.
- 7**: The 'Format' is set to 'AU\$100,000.00'.
- 8**: The 'Minimum Value' is set to '(no suitable fields)'.

At the bottom, there are 'Cancel' and 'Save' buttons. The 'Save' button is highlighted with a yellow circle.

A second Number field is required to do this calculation because we need to input the hours worked.

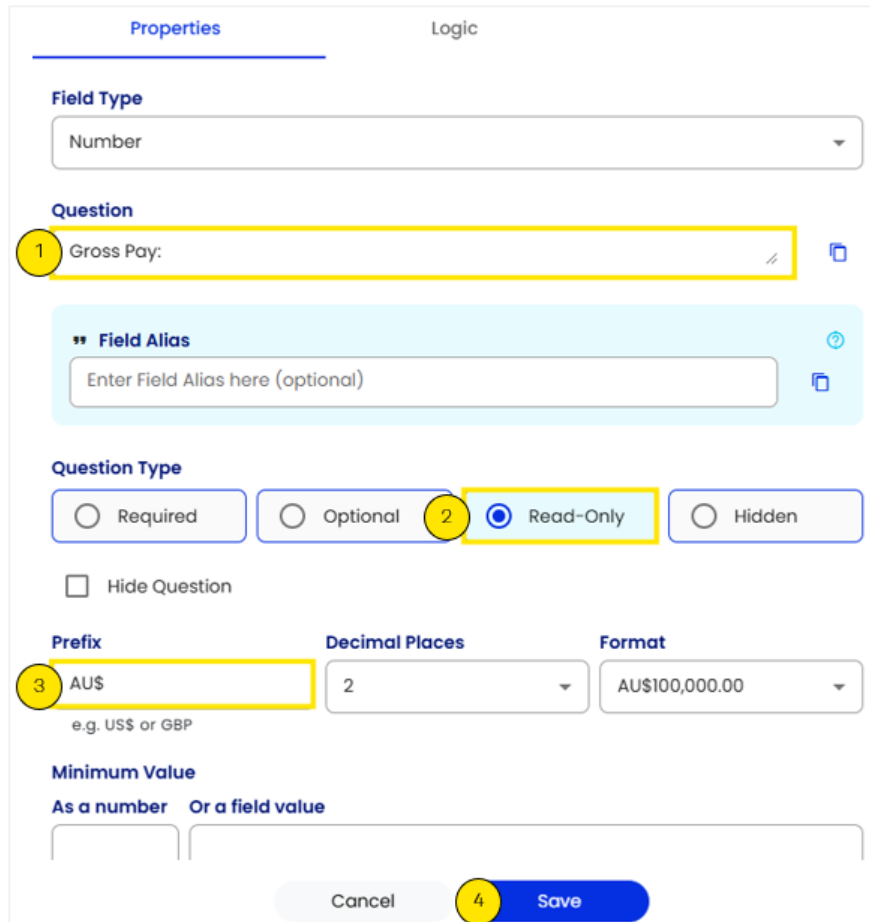
1. Add a **Question** to prompt the form filler to enter the data.
2. This **Question Type** is **Required** to complete the calculation.
3. No **Prefix** is required for hours.
4. Click **Save**.



The screenshot displays the 'Properties' tab of a question configuration interface. The 'Field Type' is set to 'Number'. The 'Question' text is 'Hours worked:'. The 'Question Type' is set to 'Required'. The 'Prefix' field is empty. The 'Decimal Places' are set to '2' and the 'Format' is '100,000.00'. The 'Minimum Value' section has two input fields, both of which are empty. At the bottom, there are 'Cancel' and 'Save' buttons. Yellow circles with numbers 1 through 4 highlight the 'Question' text, the 'Required' radio button, the 'Prefix' field, and the 'Save' button, respectively.

We can now create a calculation field, to multiply the hourly rate by the hours worked.

1. The **Question** is 'Gross Pay', which is the calculation output.
2. This **Question Type** is Read Only, as it is the calculation.
3. The **Prefix** for the output is a currency.
4. Click **Save**.



The screenshot shows the 'Properties' tab of a configuration interface. It features two main sections: 'Properties' and 'Logic'. Under 'Properties', there are several input fields and options:

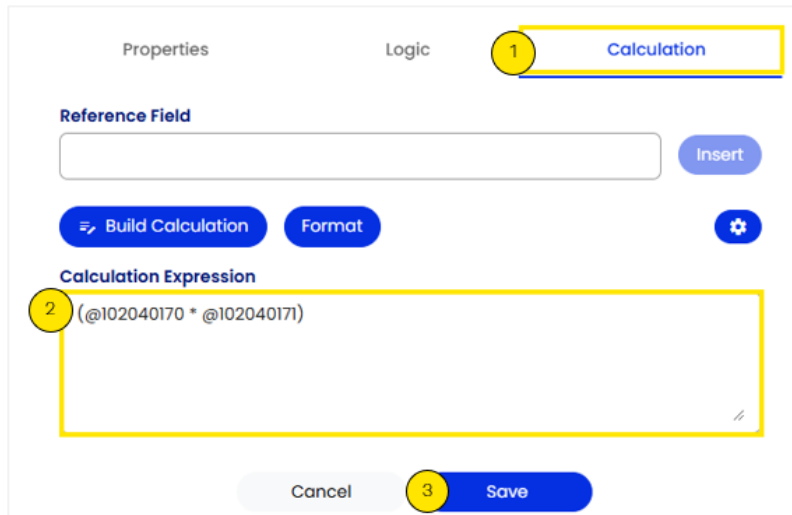
- Field Type:** A dropdown menu set to 'Number'.
- Question:** A text input field containing 'Gross Pay:' with a yellow circle '1' next to it.
- Field Alias:** A text input field with the placeholder 'Enter Field Alias here (optional)'.
- Question Type:** A group of radio buttons with options 'Required', 'Optional', 'Read-Only', and 'Hidden'. The 'Read-Only' option is selected and highlighted with a yellow circle '2'.
- Hide Question:** A checkbox that is currently unchecked.
- Prefix:** A text input field containing 'AUS' with a yellow circle '3' next to it. Below it is the text 'e.g. US\$ or GBP'.
- Decimal Places:** A dropdown menu set to '2'.
- Format:** A dropdown menu set to 'AU\$100,000.00'.
- Minimum Value:** Two input fields labeled 'As a number' and 'Or a field value', both of which are empty.

At the bottom of the form, there are two buttons: 'Cancel' and 'Save'. The 'Save' button is highlighted with a yellow circle '4'.



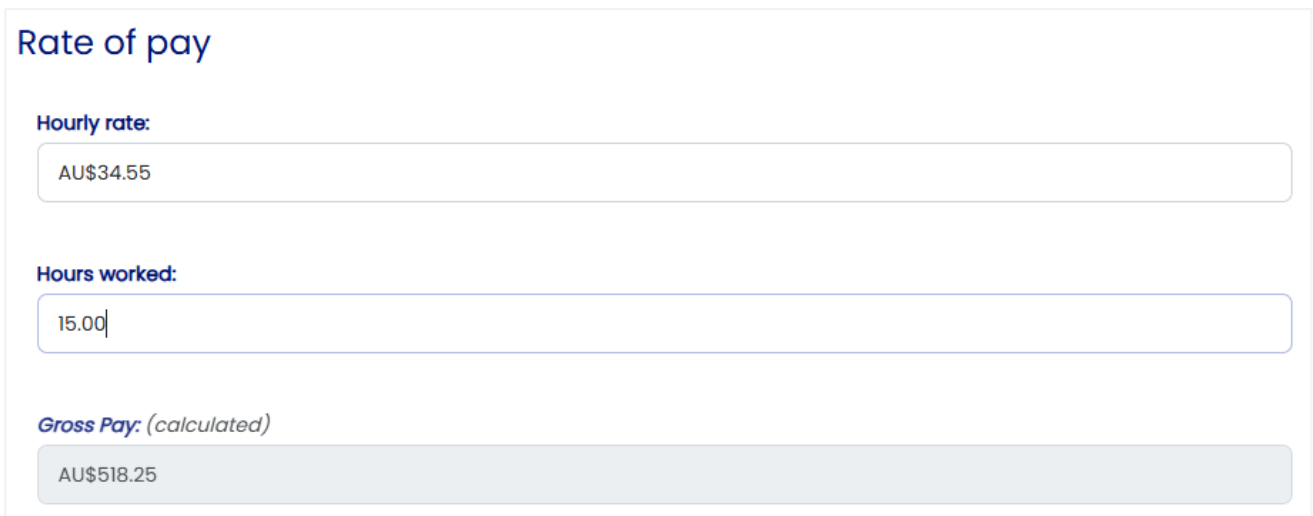
To create the calculation:

1. Open the **Calculation** tab.
2. Copy and paste the field IDs to build the calculation. Place the field IDs in parenthesis and use an asterisk to give the command to multiply the contents of the fields.
3. Click **Save**. This form can now be tested.



## 6.1 Fill in the Form

When this form is filled in, the hourly rate entered will be multiplied by the hours worked and displayed in the read-only **Gross Pay** field.



## 7. String Calculations

Smarter Drafter can perform string queries and manipulations. The following string methods are very useful for providing custom validation for users' answers and creating highly tailored field labels and rich text field guidance.

---

Note: "String" is term that means a series of characters. A name, address, email address, phone number or sentence – these are all examples of strings, because they are made up of characters.

---

To create a string calculation, we reference the field (@reference#) and apply the method code with a similar syntax. For example: @123456789.Method.

In addition, we can add a question mark (?) to indicate that the argument is optional.

Let's look at some examples:

### 7.1 IndexOf

The IndexOf() method finds a defined search term within a target string, and compares it to an index. IndexOf() will return the index of the first character of the search term's first occurrence within the target string or zero if the search term is not found. It will return zero when the search term is an empty string or a field that is unanswered or hidden by logic.

For example: A field in your form asks a form filler to list their assets. You could create an IndexOf calculation to identify if they have listed "real estate".

The final syntax would look like this: @123456789.IndexOf("real estate") = 1.

We can continue to add other asset classes, and continue the hidden field attached to the calculation would be populated by the number one if the form filler entered the words "real estate".

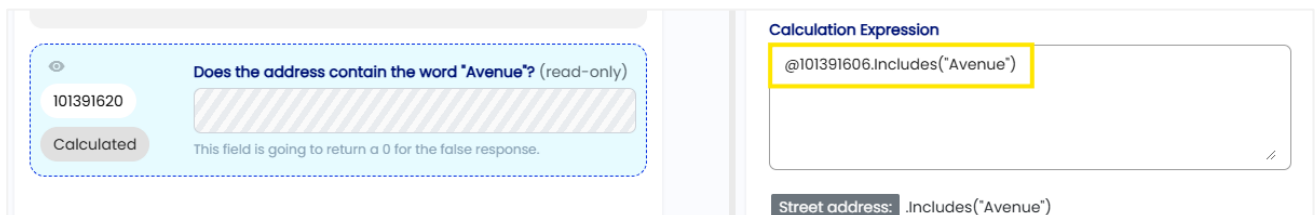
## 7.2 Includes

The Includes() method is very similar to the IndexOf() method. The only difference is that the Includes method returns 1 (representing true) if the search term is found within the target string, instead of the index of the search term.

Includes() also returns zero if the search term is not found, including when the search term is an empty string or a field that is unanswered or hidden by logic.

For example, in the calculation below, we are searching a field (ID: 101391606) which contains part of an address (Street or Avenue).

- The resulting expression: @101391606.Includes("Avenue") = 1.
- We could add a second calculation for a @101391606.Includes("Street") = 0.



The screenshot shows a form field with the ID 101391620. The field is titled "Does the address contain the word 'Avenue'? (read-only)". Below the title, there is a calculated value of 0 and a message: "This field is going to return a 0 for the false response." To the right, a "Calculation Expression" box contains the formula: @101391606.Includes("Avenue"). Below this, a label "Street address:" is followed by the formula: .Includes("Avenue").

## 7.3 CharCodeAt

The CharCodeAt() method returns a UTF-16 decimal code unit of the character at the specified index. For characters with codes between 0 and 127 UTF-16 code unit is the same as the ASCII code. For example, 'A' = 65, 'Z' = 90, 'a' = 97 and 'z' = 122.

If the index is out of range or a field hidden by logic or unanswered, the method returns null. CharCodeAt() is useful for validating text field answers when a specified format is required – for example, a specific combination of letters and numbers or a slash '/' or dash '-' in a specific position.

For example: A field (ref: 101391605) captures a first name. The name entered is Gary.

- Syntax: @101391605.CharCodeAt(1)

The output in the calculation field = 71.

Here's the full code used:

| Decimal | Binary   | Octal | Hex | ASCII | Decimal | Binary   | Octal | Hex | ASCII | Decimal | Binary   | Octal | Hex | ASCII | Decimal | Binary   | Octal | Hex | ASCII |
|---------|----------|-------|-----|-------|---------|----------|-------|-----|-------|---------|----------|-------|-----|-------|---------|----------|-------|-----|-------|
| 0       | 00000000 | 000   | 00  | NUL   | 32      | 00100000 | 040   | 20  | SP    | 64      | 01000000 | 100   | 40  | @     | 96      | 01100000 | 140   | 60  | `     |
| 1       | 00000001 | 001   | 01  | SOH   | 33      | 00100001 | 041   | 21  | !     | 65      | 01000001 | 101   | 41  | A     | 97      | 01100001 | 141   | 61  | a     |
| 2       | 00000010 | 002   | 02  | STX   | 34      | 00100010 | 042   | 22  | "     | 66      | 01000010 | 102   | 42  | B     | 98      | 01100010 | 142   | 62  | b     |
| 3       | 00000011 | 003   | 03  | ETX   | 35      | 00100011 | 043   | 23  | #     | 67      | 01000011 | 103   | 43  | C     | 99      | 01100011 | 143   | 63  | c     |
| 4       | 00000100 | 004   | 04  | EOT   | 36      | 00100100 | 044   | 24  | \$    | 68      | 01000100 | 104   | 44  | D     | 100     | 01100100 | 144   | 64  | d     |
| 5       | 00000101 | 005   | 05  | ENQ   | 37      | 00100101 | 045   | 25  | %     | 69      | 01000101 | 105   | 45  | E     | 101     | 01100101 | 145   | 65  | e     |
| 6       | 00000110 | 006   | 06  | ACK   | 38      | 00100110 | 046   | 26  | &     | 70      | 01000110 | 106   | 46  | F     | 102     | 01100110 | 146   | 66  | f     |
| 7       | 00000111 | 007   | 07  | BEL   | 39      | 00100111 | 047   | 27  | '     | 71      | 01000111 | 107   | 47  | G     | 103     | 01100111 | 147   | 67  | g     |
| 8       | 00001000 | 010   | 08  | BS    | 40      | 00101000 | 050   | 28  | (     | 72      | 01001000 | 110   | 48  | H     | 104     | 01101000 | 150   | 68  | h     |
| 9       | 00001001 | 011   | 09  | HT    | 41      | 00101001 | 051   | 29  | )     | 73      | 01001001 | 111   | 49  | I     | 105     | 01101001 | 151   | 69  | i     |
| 10      | 00001010 | 012   | 0A  | LF    | 42      | 00101010 | 052   | 2A  | *     | 74      | 01001010 | 112   | 4A  | J     | 106     | 01101010 | 152   | 6A  | j     |
| 11      | 00001011 | 013   | 0B  | VT    | 43      | 00101011 | 053   | 2B  | +     | 75      | 01001011 | 113   | 4B  | K     | 107     | 01101011 | 153   | 6B  | k     |
| 12      | 00001100 | 014   | 0C  | FF    | 44      | 00101100 | 054   | 2C  | ,     | 76      | 01001100 | 114   | 4C  | L     | 108     | 01101100 | 154   | 6C  | l     |
| 13      | 00001101 | 015   | 0D  | CR    | 45      | 00101101 | 055   | 2D  | -     | 77      | 01001101 | 115   | 4D  | M     | 109     | 01101101 | 155   | 6D  | m     |
| 14      | 00001110 | 016   | 0E  | SO    | 46      | 00101110 | 056   | 2E  | .     | 78      | 01001110 | 116   | 4E  | N     | 110     | 01101110 | 156   | 6E  | n     |
| 15      | 00001111 | 017   | 0F  | SI    | 47      | 00101111 | 057   | 2F  | /     | 79      | 01001111 | 117   | 4F  | O     | 111     | 01101111 | 157   | 6F  | o     |
| 16      | 00010000 | 020   | 10  | DLE   | 48      | 00110000 | 060   | 30  | 0     | 80      | 01010000 | 120   | 50  | P     | 112     | 01110000 | 160   | 70  | p     |
| 17      | 00010001 | 021   | 11  | DC1   | 49      | 00110001 | 061   | 31  | 1     | 81      | 01010001 | 121   | 51  | Q     | 113     | 01110001 | 161   | 71  | q     |
| 18      | 00010010 | 022   | 12  | DC2   | 50      | 00110010 | 062   | 32  | 2     | 82      | 01010010 | 122   | 52  | R     | 114     | 01110010 | 162   | 72  | r     |
| 19      | 00010011 | 023   | 13  | DC3   | 51      | 00110011 | 063   | 33  | 3     | 83      | 01010011 | 123   | 53  | S     | 115     | 01110011 | 163   | 73  | s     |
| 20      | 00010100 | 024   | 14  | DC4   | 52      | 00110100 | 064   | 34  | 4     | 84      | 01010100 | 124   | 54  | T     | 116     | 01110100 | 164   | 74  | t     |
| 21      | 00010101 | 025   | 15  | NAK   | 53      | 00110101 | 065   | 35  | 5     | 85      | 01010101 | 125   | 55  | U     | 117     | 01110101 | 165   | 75  | u     |
| 22      | 00010110 | 026   | 16  | SYN   | 54      | 00110110 | 066   | 36  | 6     | 86      | 01010110 | 126   | 56  | V     | 118     | 01110110 | 166   | 76  | v     |
| 23      | 00010111 | 027   | 17  | ETB   | 55      | 00110111 | 067   | 37  | 7     | 87      | 01010111 | 127   | 57  | W     | 119     | 01110111 | 167   | 77  | w     |
| 24      | 00011000 | 030   | 18  | CAN   | 56      | 00111000 | 070   | 38  | 8     | 88      | 01011000 | 130   | 58  | X     | 120     | 01111000 | 170   | 78  | x     |
| 25      | 00011001 | 031   | 19  | EM    | 57      | 00111001 | 071   | 39  | 9     | 89      | 01011001 | 131   | 59  | Y     | 121     | 01111001 | 171   | 79  | y     |
| 26      | 00011010 | 032   | 1A  | SUB   | 58      | 00111010 | 072   | 3A  | :     | 90      | 01011010 | 132   | 5A  | Z     | 122     | 01111010 | 172   | 7A  | z     |
| 27      | 00011011 | 033   | 1B  | ESC   | 59      | 00111011 | 073   | 3B  | ;     | 91      | 01011011 | 133   | 5B  | [     | 123     | 01111011 | 173   | 7B  | {     |
| 28      | 00011100 | 034   | 1C  | FS    | 60      | 00111100 | 074   | 3C  | <     | 92      | 01011100 | 134   | 5C  | \     | 124     | 01111100 | 174   | 7C  |       |
| 29      | 00011101 | 035   | 1D  | GS    | 61      | 00111101 | 075   | 3D  | =     | 93      | 01011101 | 135   | 5D  | ]     | 125     | 01111101 | 175   | 7D  | }     |
| 30      | 00011110 | 036   | 1E  | RS    | 62      | 00111110 | 076   | 3E  | >     | 94      | 01011110 | 136   | 5E  | ^     | 126     | 01111110 | 176   | 7E  | ~     |
| 31      | 00011111 | 037   | 1F  | US    | 63      | 00111111 | 077   | 3F  | ?     | 95      | 01011111 | 137   | 5F  | _     | 127     | 01111111 | 177   | 7F  | DEL   |

## 7.4 SubStr

The SubStr() method extracts a substring of up to a defined number of characters from the target string starting at a certain point. If the starting point is negative, the counting starts from the end of the string.

The length of the string is optional and if it is:

- not provided,
- exceeds actual string length, or
- is a field hidden by logic or unanswered.

All characters from the starting point to the end of the target string are returned.

For example: This example is going to return characters 10-12 of the address. The address is:

Level 8, 100 George Street, Parramatta NSW 2150

- Syntax: @101391606.SubStr(10, 3)
- Output: 100, because the string starting point is 10, and the string length is 3.

## 7.5 Left/Right

The **Left()** method returns the first number of characters of a string – as in, it starts on the **left**. If the number of characters is an unanswered field or is less than zero it's treated as if it was zero. If it exceeds the string's length, it's treated as if it was equal to the length.

For example: a field (id: 101391606) has captured an address and we want to return the first 7 characters of the address. The address is: **Level 8, 100 George Street, Parramatta NSW 2150**

- Syntax: @101391606.Left(7)
- Output: Level 8

The **Right()** method returns the last number of characters of a string – it starts on the **right**.

For example: a field (id: 101391606) has captured an address and we want to extract the last 4 characters of the address. The address is: **Level 8, 100 George Street, Parramatta NSW 2150**

- Syntax: @101391606.Right(4)
- Output: 2150

## 7.6 Length

Returns the strings length, i.e. the number of characters that make up the string.

For example: a field (id: 101391606) has captured an address and we want to return the length of the address. The address is: **Level 8, 100 George Street, Parramatta NSW 2150**

- Syntax: @101391606.Length
- Output: 47

## 7.7 Concatenation

Strings can be concatenated (joined or linked together) with the '+' operator.

For example, a text field captures a place of work (id: 101033490) and "Friendly Lawyers" is entered. It can be joined with a text field that captures an occupation (id: 101033491) and "Lawyer" is entered.

- Syntax: @101033490+@101033491
- Output: Friendly Lawyers Lawyer

If only one operand is a string, all other operands are converted to strings.