The purpose of this document is to show you that you can use DocMergy using the basic native HubSpot properties based on a **contact trigger.**

You can download this document and put into the DocMergy to see how it works. If some tokens are not working and appear as they are written in code just try rewrite it out

**Document structure**

* Basic Tokens
* Get Associated Objects
* Search by Unique IDs
* If Branches
* If Block Branches
* Dynamic Images
* Dynamic Hyperlinks

The properties you will need to have filled on a contact are:

First Name
Last Name
Last Contacted
Company Name
Country/Region
City
Street Address
Close Date
Create Date
Annual Revenue
Phone

A Deal
 Deal Name
 Note on the Deal
 Create Date of the Note

Line Items
 Line Item Name
 Line Item Description

Website URL: you will need to create a custom property for image URL and then search it up on the token generator to get the token to insert here.

The property should be called: ‘Profile Picture URL’ and the internal name will be: ‘profile\_picture\_url’

**Basic Tokens**

**Basic native HubSpot property tokens**

First name: {{ object.firstname }}
Last name: {{ object.lastname }}
First name last name: {{ object.firstname }} {{ object.lastname }}
First name last name upper case: {{ object.firstname.value.upper() }} {{ object.lastname.value.upper() }}
Last contacted: {{ object.notes\_last\_contacted.value.strftime ("%A, %d %B %Y") }}

Company Name: {{ object.company }}
Country/Region : {{ object.country }}
City: {{ object.city }}
Street Address: {{ object.address }}

**1. Basic Tokens with styling**
First name upper case: {{ object.firstname.value.upper() }}
First name and last name in bold: **{{ object.firstname }} {{ object.lastname }}**

**2. Basic Tokens with formatting (Date Formatting & Token Generator)**

Create Date: {{ object.createdate }}
Create Date (eg. 09 April 2025): {{ object.createdate.value.strftime("%d %B %Y") }}
Create Date (eg. 09/04/2025): {{ object.createdate.value.strftime("%d/%m/%Y") }}
 **3. Basic Tokens in text boxes**

Annual Revenue
{{ object.annualrevenue }}

Annual Revenue

{{ object.annualrevenue }}

Phone

{{ object.phone }}

Contact First Name

{{ object.firstname }}

Phone

{{ object.phone }}

First Name

{{ object.firstname }}

**4.** **Basic Tokens in header & footer\*\*\***

To {{ object.firstname }} {{object.lastname }}

* Goes in the heading section

**Get Associated Objects**

**1. Get singular version**

* You will need to associate a deal to a contact and vice versa, you can do this manually
* It shows the name of the first deal associated with the current contact.

{{ get\_associated\_objects("contact", object.hs\_object\_id, "deal")[0].dealname }}

The function get\_associated\_objects takes 4 arguments

* "contact": is the source object of the association
* object.hs\_object\_id: the id of the source object, here it is the enrolled object ID
* "deal" the associated object type you want to get data from
* The last argument is "order\_by" and is the property you want to use for ordering the list of "deals". You can get more info here <https://support.integrationglue.com/docmergy-how-to-get-data-from-associated-objects>

the  [0] tells DocMergy to only grab the first associated deal you can here use [1] if you want the second one or [-1] if you want the last one
dealname is the internal property name of the "deal" you want to display
and finally the {{ }} around it are telling DocMergy to render the property in the document

**2. For loop version**

* This loops through all notes on the first deal associated with the contact, and shows each note’s content and creation date.
* You can create multiple notes to see the tables be filled

|  |  |
| --- | --- |
| Title | Description |
| {% for n in get\_associated\_objects("deal", get\_associated\_objects("contact", object.hs\_object\_id, "deal")[0].hs\_object\_id, "notes") %} |
| {{ n.hs\_note\_body}} | {{ n.hs\_createdate }} |

**3. For loop images**

* This code loops through the line items on the first deal linked to the current contact and displays each item's name, description, image and image text.
* You can create multiple contacts to see the tables be filled

**JPEG/JPG**

|  |  |  |
| --- | --- | --- |
| Title | Description | Image |
| {% for li in get\_associated\_objects("deal", get\_associated\_objects("contact", object.hs\_object\_id, "deal")[0].hs\_object\_id, "line\_items") %} |
| {{ li.name }} | {{ li.description }} | This Image: **{{li.image_url_jpg}}** {{li.image\_text}} {{ li.name }}  |

**Search By Unique IDs**

**1. Get singular version**

* This shows the first name of the contact with the HubSpot ID: 80716298250

Search for the unique ID of and show first name {{ search\_by\_unique\_ids(“contact”,[ 80716298250 ])[0].firstname }}

**2. For loop version**

* This loops through a list of contacts (looked up by their IDs) and displays their first name, last name, and email address.

|  |  |  |
| --- | --- | --- |
| First name | Last name | email |
| {% for c in search\_by\_unique\_ids(“contact”,[ 80716298250, 86257912096 ]) %}  |
| {{ c.firstname }} | {{c.lastname}} | {{c.email}} |

**3. For loop images**

* This loop grabs two specific contacts by their HubSpot IDs and displays each person's name and converting the profile picture URL to a visible image

|  |  |  |
| --- | --- | --- |
| First name | Last name | Profile Picture |
| {% for c in search\_by\_unique\_ids(“contact”,[ 80716298250, 86257912096 ]) %}  |
| {{ c.firstname }} | {{c.lastname}} | {{ c.profile_picture_url }} |

For looping images you will need to create a property with a public URL linking to an image. You can upload an image to HubSpot files, get the URL and put it in the property you made.

You will then need to create the custom relevant custom token which is associated to the property you made eg. Object profile picture URL and then put that in the ALT text for the image

If you right click on the image you need to insert what the token would be for the custom property you made, except “object.” It needs to be “c.profile\_picture” so its taking the other contacts URL and not the contact that is enrolled.

**If Branches**

**1. Basic If Branch**

**What it does:**

* If the firstname property exists and has a value → it will say Hello {{Firstname}}!
* If firstname is blank or missing → it will say Hello there!

{%if object.firstname%} Hello {{ object.firstname }}! {%else%} Hello there! {%endif%}

**2. If Else Branch**

* This code checks if the first deal associated with the current contact has a deal name, and if it does, it displays it — otherwise, it shows “No deal name found.

{{ get\_associated\_objects("contact", object.hs\_object\_id, "deal")[0].dealname }}

{% if get\_associated\_objects("contact", object.hs\_object\_id, "deal")[0].dealname %} Associated Deal Name: {{ get\_associated\_objects("contact", object.hs\_object\_id, "deal")[0].dealname }} {% else %}No deal name found {% endif %}

**3. If Elif** **Branch**

{% if object.country.value =="New Zealand" %} Kia ora! Looks like you're based in New Zealand {% elif object.country.value == "Australia" %} G'day! You're located in Australia {% endif %}

* If the contact’s country is "New Zealand" →

➜ Kia ora! Looks like you're based in New Zealand.

* If it’s "Australia" →
➜ G'day! You're located in Australia.
* If it’s something else or blank → nothing shows (no else block included).

**4. If Elif Else Branch**{% if object.firstname == "John" %}Hello, John! We're happy to assist you today{% elif object.firstname == "Jane" %}Hello, Jane! We’re excited to help you with your inquiry{% else %}Hello, {{ object.firstname }}! Thanks for reaching out to us {% endif %}

How it works

**If**: If the contact's first name is "John", it will display:
 Hello, John! We're happy to assist you today

**Elif**: If the contact's first name is "Jane", it will display:
 Hello, Jane! We’re excited to help you with your inquiry

**Else:** If the contact's first name is any other name, it will display:
 Hello, [First Name]! Thanks for reaching out to us.

If its any other first name eg. Alex, it will display
 Hello, Alex! Thanks for reaching out to us.

The name is dynamic but everything after will remain the same “Thanks for reaching out to us”. John and Jane have their own text to display

**If Block Branch**

**1. Examples for normal text**

* If the contact’s first name is Sam, the output would be: “Hello Ben\*, thanks for reaching out!
* If the first name is blank, the whole block (including the greeting) is skipped entirely.

{% block if object.firstname %}

Hello {{ object.firstname }}, thanks for reaching out!

{% end block %}

**2. Examples for in tables**

* If the contact has a first name, it displays their full name in a text box

|  |  |
| --- | --- |
| Contact  | Email |
| Someone Interesting  | SomeoneInteresting@gmail.com |
| {% block if object.firstname %}{{ object.firstname }} {{ object.lastname }}{% end block %} | JohnInteresting@gmail.com |

**3. Examples for deleting images, tables and text boxes**

* ****If the token contains no value, delete everything between
and
	+ In this case the property is empty
* If the token does contain a value, then keep everything in between.

{% block if object.contact\_owner %}

|  |  |
| --- | --- |
|  |  |

This should be deleted

**JASLKFHKLASJHF**

****

|  |  |
| --- | --- |
| This should be deleted | Description 2 |
| 154998721546 number | 03:03AM. 31 Jan 24 |
| 154998721546 number | 03:03AM. 31 Jan 24 |
| 154998721546 number | 03:03AM. 31 Jan 24 |

{% end block %}

**Dynamic Images**

* Dynamic images use a placeholder but will take an image URL associated with the enrolled object

**Eg.** Display Profile Picture

* Eg. With the custom property you made for the profile picture, put the token in the ALT text

****

**Dynamic Hyperlinks**

* Dynamic hyperlinks use a placeholder of any text of your choice but will take the website associated with the enrolled object and change according to the enrolled object.
* If the contact has a property with a website URL in it then the Dynamic Hyperlink grabs that link and puts it in the text

Website URL (Edit Link): [This is Dynamic Link to the Website URL](%7B%7B%20object.website%20%7D%7D)