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

You can download this document and put into the DocMergy to see how it works.

You will just need to pre fill the properties below which can just be made up

**Document structure**

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

Required properties that need to be prefilled in order for this work

Deal Name  
Deal Amount  
Deal Beginning Date  
Deal Close Date   
Deal Description

A Contact   
 Contact First Name  
 Contact Email  
Have 2 contacts associated with a deal

Line Item Name  
Line Item Description  
Need to create a custom property called: ‘Product Image URL’ with the internal name being ‘product\_image\_url’ and the token being {% raw %} {{ object.product\_image\_url }} {% endraw %}

Have a deal in the Appointment Scheduled Stage

Have Deal Collaborator empty (no value in the property)

**Basic Tokens**

**Basic native HubSpot property tokens**

Deal name: {{ object.dealname }}  
Deal name in bold: **{{ object.dealname }}**  
Deal name in capitals: {{ object.dealname.value.upper() }}  
Deal Amount: {{ object.amount }}  
Deal Beginning Date: {{ object.deal\_beginning\_date }}  
Deal Close Date: {{ object.closedate }}  
Deal Description: {{ object.description }}

**1. Basic Tokens with styling**  
Deal name: {{ object.dealname }}  
Deal name in bold: **{{ object.dealname }}**  
Deal name in bold upper case: **{{ object.dealname.value.upper() }}**

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

Deal Close Date normal: {{ object.closedate }}  
Deal Close Date (eg. 13 April 2025: {{ object.closedate.value.strftime("%d %B %Y") }}  
Deal Close Date(eg. Sun, 13 April 25: {{ object.closedate.value.strftime("%a, %d %b %y") }}  
Close Date (eg. 13/04/2025): {{ object.closedate.value.strftime(“%d/%m/%Y”) }}

**3. Basic Tokens in text boxes**

Deal amount  
{{ object.amount }}

Deal Amount

{{ object.amount }}

Deal Beginning Date: {{ object.deal\_beginning\_date }}

Deal Name

{{ object.dealname }}

Deal Begin Date: {{ object.deal\_beginning\_date }}

Deal name   
{{ object.dealname }}

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

This deal is: {{ object.dealname }}

* Double click the heading or footing section and place the token there

**Get Associated Objects**

**1. Get singular version**

* The code gets the **first contact associated with the deal** and returns their **first name**

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

The function get\_associated\_objects takes 4 arguments

1. "deal": is the source object of the association
2. object.hs\_object\_id: the id of the source object, here it is the enrolled object ID
3. "contact" the associated object type you want to get data from
4. 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
* Firstname is the internal property name of the "contact" you want to display
* The {{ }} around it are telling docmergy to render the property in the document

**2. For loop version**

* For every contact associated with the current deal, show their first name and email address

|  |  |
| --- | --- |
| First Name | Email |
| {% for n in get\_associated\_objects("deal", object.hs\_object\_id, "contact") %} | |
| {{ n.firstname }} | {{ n.email }} |

**3. For loop images**

* This code loops through all line items associated with a deal and displays the line item name, description as well as the image and image text

|  |  |  |
| --- | --- | --- |
| Title | Description | Image |
| {% for li in get\_associated\_objects("deal", object.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 deal name of the with the HubSpot ID: 32226481156

Search for the unique ID of and show me the deal name: {{ search\_by\_unique\_ids(“deal”,[ 32226481156 ])[0].dealname }}

**2. For loop version**

* Loop through two specific deals, identified by their HubSpot object IDs (32226481156 and 35640046409), getting the deal name, deal amount and deal close date and store each one in the variable c for use inside the loop.

|  |  |  |
| --- | --- | --- |
| Deal Name | Deal Amount | Deal Close date |
| {% for c in search\_by\_unique\_ids(“deal”,[ 32226481156, 35640046409 ]) %} | | |
| {{ c.dealname }} | {{c.amount}} | {{c.closedate}} |

* search\_by\_unique\_ids("deal", [...]) pulls full data for the listed deal IDs.
* The for loop goes through each one and refers to it as c.
* Inside the loop, you can access things like dealname, amount, ect

**3. For loop images**

* This loop grabs two specific deals by their HubSpot IDs and displays each deals name, amount and converting the ‘product picture URL’ to a visible image
* You will need to create a custom property for this called: ‘Product Image URL’ with the internal name being ‘product\_image\_url’
* Upload some images to HubSpot files, grab the URL and paste it into the property on the two deals. To see the demo get two random images and put each URL on two different deals.
* Put the token {% raw %} {{ c.product\_image\_url }} {% endraw %} in the image alt text below. It needs to be “c.” so its taking the other deals URL and not just the deal that is enrolled

|  |  |  |
| --- | --- | --- |
| Deal Name | Deal Amount | Profile Picture |
| {% for c in search\_by\_unique\_ids(“deal”,[ 32226481156, 35640046409 ]) %} | | |
| {{ c.dealname }} | {{c.amount}} | {{c.product_image_url}} |

**If Branches**

**1. Basic If Branch**

* If dealname has a value, it shows the message.
* If it's blank or missing, it shows nothing at all

{% if object.dealname %} Hello! You’re working on the {{object.dealname}} deal. {% endif %}

**2. If Else Branch**

* This code checks if the **deal has a name**. If it does, it shows a message with the deal name; if not, it shows a default message saying the deal doesn't have a name yet.

{% if object.dealname %} Hello! You’re working on the {{object.dealname}} deal. {% else %} Hello! This deal doesn’t have a name yet. {% endif %}

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

* This code checks the dealstage of a deal
* If it's "Appointment Scheduled," it shows one message;
* If it's "Closed Won," it shows a different message
* Wont show anything if neither of those are recognized

{% if object.dealstage.value == "appointmentscheduled" %} This deal is in the Appointment Scheduled stage {% elif object.dealstage.value == "closedwon" %} This deal has been won! {% endif %}

**4. If Elif Else Branch.**

* This code checks the dealstage of a deal
* If it's "Appointment Scheduled," it shows one message;
* If it's "Closed Won," it shows a different message
* If its anything else it will show: This deal is in a different stage

{% if object.dealstage.value == "appointmentscheduled" %} This deal is in the Appointment Scheduled stage {% elif object.dealstage.value == "closedwon" %} This deal has been won! {% else %} This deal is in a different stage {% endif %}

**If Block Branch**

**1. Examples for normal text**

* If the dealname property **has a value**, it shows the message. If it's empty or missing, nothing is shown.

{% block if object.dealname %}

Hello! You’re working on the {{object.dealname}} deal.

{% end block %}

**2. Examples for in tables**

* If the dealname property exists and has a value, it shows the deal’s name followed by its description.
* If dealname is blank, nothing is shown.

|  |  |
| --- | --- |
| Deal Name and Description | Deal Start Date |
| Purchase Agreement – Purchase agreement for the product | 10 January 2023 |
| {% block if object.dealname %}  {{ object.dealname }} - {{ object.description }}  {% end block %} | 18 August 2024 |

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

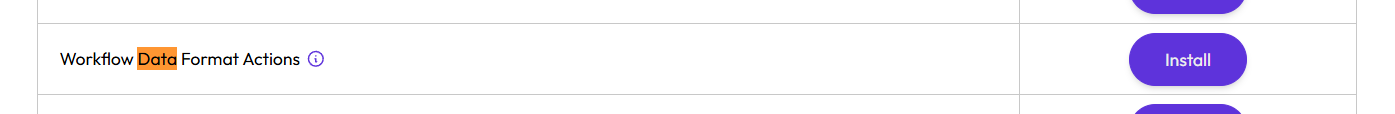
* ****The code is setup so If the token contains no value, delete everything between   
  and
  + If the token **does** contain a value, then keep everything in between.
  + In this case the property for ‘Deal Collaborator’ is **empty** so when you run it through DocMergy there will be nothing shown

{% block if object.hs\_all\_collaborator\_owner\_ids %}

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

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 %}

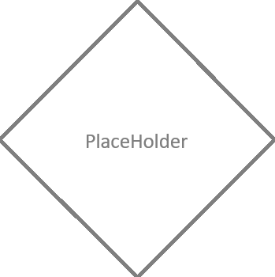
**Dynamic Images**

* Dynamic images use a placeholder but will take an image URL on the custom property on the enrolled object

**Eg.** Display Product Picture

* Eg. With the custom property you made for the product image, put the token in the ALT text as seen below, this allows it to change dynamically

A screen shot of a computer

AI-generated content may be incorrect.

**Dynamic Hyperlinks**

* Dynamic hyperlinks use a placeholder of any text of your choice but will take the property with a URL (could be a website) associated with the enrolled object and change according to the enrolled object.
* If the contact has a property with a URL in it then the Dynamic Hyperlink grabs that link and puts it in the text due to the token in the hyperlink
* This URL will take you to your product image on the deal
* ****

URL (Edit Link): [This is Dynamic Link to the Website URL](file:///C:\Users\nic.clement_integrat\Downloads\%7b%7b%20object.product_image_url%20%7d%7d)