1. Is able to replace MERGEFIELDS with the appropriate data.

**Instruction**: Add a Mail Merge field that will be replaced with real information.

First Name: «first\_name» Last Name: «last\_name»

1. It replaces MERGEFIELD with one or more paragraph, and can overflow across pages:

**Instruction**: Add a MERGEFIELD at the location where you want the information to appear.

«a\_long\_paragraph»

1. Can check checkboxes:

**Instructions**:

* 1. Use the Developer toolbar to add a checkbox;
  2. Double click, or right click the checkbox, and select properties.
  3. Add a unique token in the ‘bookmark’ field; which will be the key used in the input data stream to set the ‘checked’ state of the checkbox.

Check Me

Leave me unchecked

1. Can populate table rows with a sequence:

**Instructions:**

* 1. Create a table and format it
  2. In whichever row you want to start the sequence, insert three rows – the first and third should have ***start:****unique\_token* and***end:****unique\_token* MAILMERGE fields in the first column, and nothing in the rest of the row. The second row, between the **start** and **end** token rows, will be repeated once per item in the sequence, and should have mail merge fields that will be replaced with the given attribute of each item.

|  |  |  |  |
| --- | --- | --- | --- |
| **Meal** | **Drink** | **Appetizer** | **Dessert** |
| «start:meals» |  |  |  |
| «meal» |  | «appetizer» | «dessert» |
|  | «drink» |  |  |
| «end:meals» |  |  |  |

1. Can repeat a block with a sequential index:

**Instructions:**

* 1. Create the block that you to repeat
  2. Inside the block, define mail MERGEFIELDs that will be repeated with different information.
  3. Around the block, insert a ***start:****unique\_token* and***end:****unique\_token* MAILMERGE markers to indicate the boundaries of the block to repeat.
  4. The final result should remove the boundary markers.

«start:owners»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«last\_name», «first\_name»

Repeat including this line.

«end:owners»

«start:owners»

Yay look it’s «first\_name»

«end:owners»

1. Can access nested values in data set:  
   **Instruction:** Define a Mail Merge field and use dots (“.”) to access deeper levels of nesting in the dataset.  
     
   First Level: ***«embedded.first\_level»*** Second Level: «embedded.two\_levels.second\_level»
2. Can include and exclude blocks conditionally:  
   Instruction: Place ***if:****unique\_token* and ***end\_if:****unique\_token* (or ***unless:****unique\_token* and ***end\_unless:****unique\_token*) tags around the blocks you want to include (or exclude) if that variable exists in the dataset.  
     
   «if:exceptions»This text will not be shown, but «end\_if:exceptions»This text will.  
   «unless:exceptions»  
   This text will also be shown, since it’s the inverse conditional.«if:reviewed» But this won’t.  
   «end\_unless:exceptions»
3. Can create lists using sequences:  
   **Instruction:** Create a list (numbered or bulleted), and make one of the elements a ***start****:unique\_token* and one an ***end:****unique\_token*; in between, place your merge fields.

Here’s a list:

* 1. «start:grocery\_list»
  2. Need quantity **«quantity»** of «product»
  3. «end:grocery\_list»

1. Can create a sequence from an array of primitives:  
   **Instruction:** If you have an array of primitives (strings, integers, etc) in your dataset, you can use a regular sequence, but within the sequence tags, use the keyword *item* for the mergefield. You can also override this with an “as” filter as shown below.  
     
   The following people are invited: «start:invitees»«item», «end:invitees»and you.  
   They need to bring: «start:potluck\_dishes»Tasty «end:potluck\_dishes»!

There should be nothing populated below, since the conditional will exclude the header as well as the list itself:

«if:vegetables»

Here’s a list:

1. «start:vegetables»
2. A vegetable is «item»
3. «end:vegetables»