Loops - display multi line/element data

Use screen elements loops to display array results of filters and API requests on an app screen. A loop can iterate on a single key/value, like - a name; or on a multiple ones, like - a name, date of birth, avatar picture, etc.

Common example of such arrays is a list of items in a shopping cart or a list of payments.

The platform allows to create loops and display results over:

- DATA FROM FILTERS
- DATA FROM OPERATIONS
- DATA FROM LISTS OF TABLES

In both these examples we will take the following steps:

- Set up data source: Filter or Operation
- Set up loop that will parse the data from the data source
- Call the loop reference and extract its' data into an app's screen

Before proceeding, make sure you know, how basic adding elements in Constructor works, PLUS:

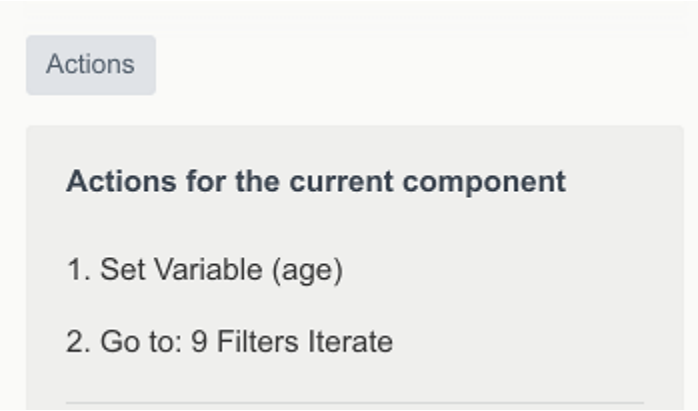
- Hashtags, as this section is heavy on referencing data points,
- Automatic Data Filters, as Loops need a source of data and Filters are the most commonly used one.

DATA FROM FILTERS

At the Filter's section we have created **AgeEmail** filter that shows users with Gmail account & with the age less than specified. Now, we create a Loop to display Name and Age of all the filtered records.

Steps to take:

1. Navigate to **Constructor**
2. Create new screen by clicking **Add New Screen** at the bottom of the platform
3. Drag and Drop **Text Input** element
4. Set its' properties:
    - **Backendname** as *Age*
    - **Default Value** as *#Variable:age#*
5. Drag and Drop **Button** element
6. Set its' properties:
    - **Backendname** as *Button*
    - **Caption Button** as *Less than Age #Backendname:Age#*
7. Click **Actions** button and **Add new action** from drop down:
    - **Set variable** with the **Name:** *Age* and **Value:** *#Backendname:Age#*. Click **Save**
    - Add **Go to** action. Select **Screen** and in **Choose screen** drop down select the screen you are on now. This will reload the page. Otherwise the filter results will not be updated.



8. Drag and Drop **Text** element to the screen

9. Set it's text: *Filtered Data:*

10. Drag and Drop another Text element

11. Enable looping for that element - find **LOOP** in the right hand side menu and check **Enabled** box

12. Set up **Data Source:** *#ObjectsFilter:AgeEmail:Data#,* where *AgeEmail* is the name of our Filter

13. Enter **Loop Name:** *AgeEmailLoop, it need a name to be referenced*

14. Go to **Text** field of the element on the app screen, and set up what will be displayed by pulling the required Object columns from the loop:

    a) to display Name use *Employee:#Loop:AgeEmailLoop:objects@Name#*

    b) to display Age use *Age:#Loop:AgeEmailLoop:objects@Age#*

WHERE

#Loop:LOOPNAME:objects@COLUMN-NAME#

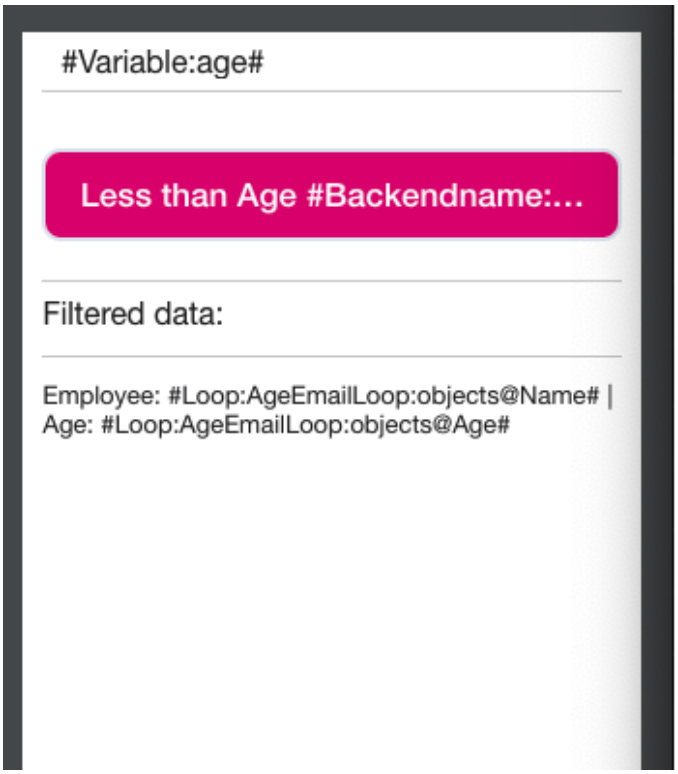
- **#LOOP** - points that data must be taken from a Loop and iterated
- **LOOPNAME** - sets a name of Loop to be used as source
- **Objects@column-name** - points at which data point to pull from the Object to display

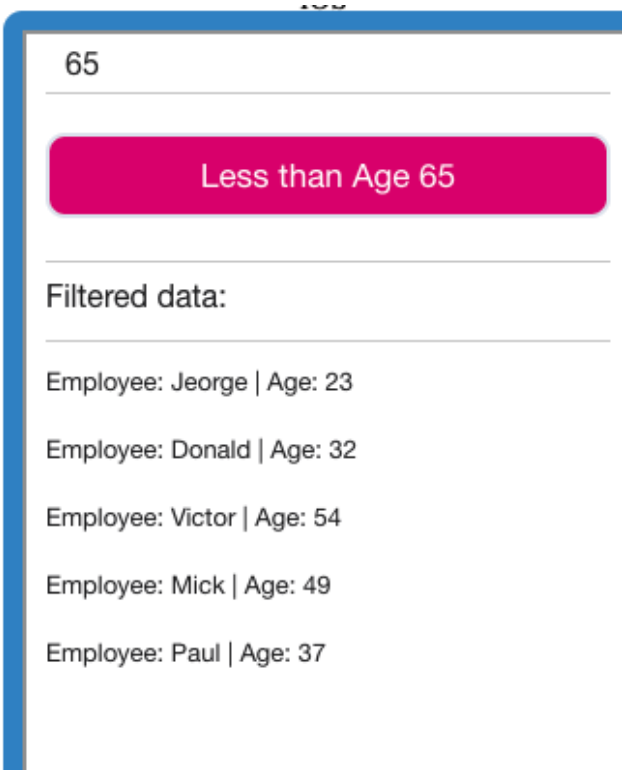
Other properties that can be referenced are:

- backend@COLUMN_NAME - for Events
- statuses@COLUMN_NAME - for Statuses of Events

15. Click **Save Screen**

16. Click **Preview** button to launch app from the screen to check it works.





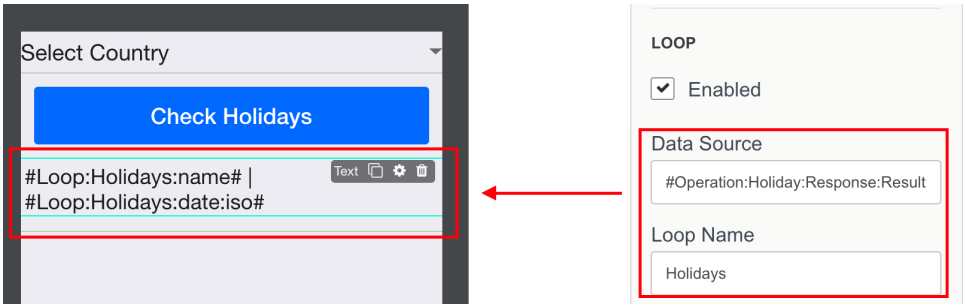
You can have a look at how it's setup in **Demo App** in your account, screen - **Filters Iterate**.

DATA FROM OPERATIONS

We have a screen, where a user can select a country from the drop down and see its' public holidays. For this example, we have pre-created API Operation to the 3rd party service.

Steps to take:

1. Drag and Drop a **Select** element
2. Use **Add Option** and create a list of countries:
    a. **Label** = *USA* **Value** = *US*
    b. **Label** = *Canada* **Value** = *CA*
    c. **Label** = *Brazil* **Value** = *BR*
3. Drag and Drop Button element
4. Set it's properties:
    a. **Backendname** = *holidaysButton*
    b. **Caption Button** = *Check Holidays*
5. Click **Actions,** select **Execute Operations** and add pre-created **Holiday** operation
6. Drag and Drop Text element
7. Enable looping for that element: find **LOOP** in the right hand side menu and check **Enabled** box



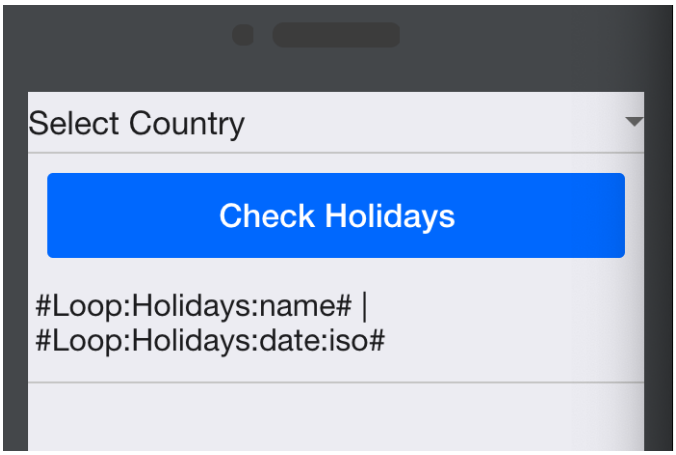
8. Set up **Data Source** for the loop: *#Operation:Holiday:Response:Result:0:response:holidays#*
9. **Name** the loop as *Holidays*
10. Go to **Text** field of the element and set up what will be displayed by pulling the required array elements from the loop (OPERATION):
    a. *#Loop:Holidays:name#*
    b. *#Loop:Holidays:date:iso#*

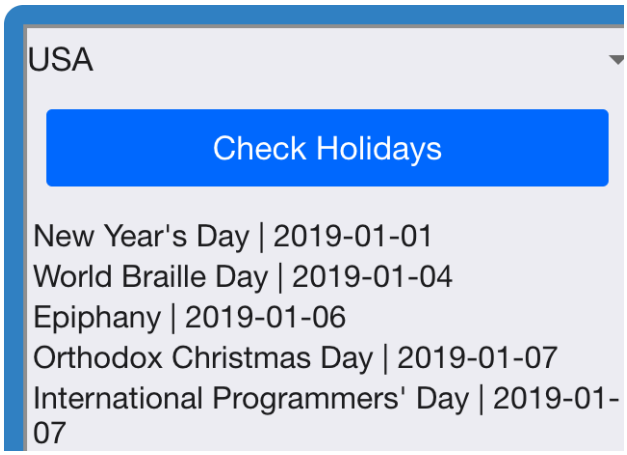
WHERE

Referencing INTERNAL data points is different compared to EXTERNAL or API delivered data:

- We only use at sign "@" to reference INTERNAL, such as Object's properties - *#Loop:LOOPNAME:**objects@COLUMN-NAME**#*
- When referencing Operations or external APIs, we use colons only - #Loop:LOOPNAME:ARRAY_KEY_1:ARRAY_KEY_N#), where RED is required and:
    - Loopname - references, which Loops to use as source,
    - Array_Key_1 - name of a key in array, returned by an Operation, to extract value from
    - Array_Key_N - if the needed Key contains another "internal" array, reference that "nested" array's keys with a : colon

11. Click **Preview** button to launch app from the screen to check it works.





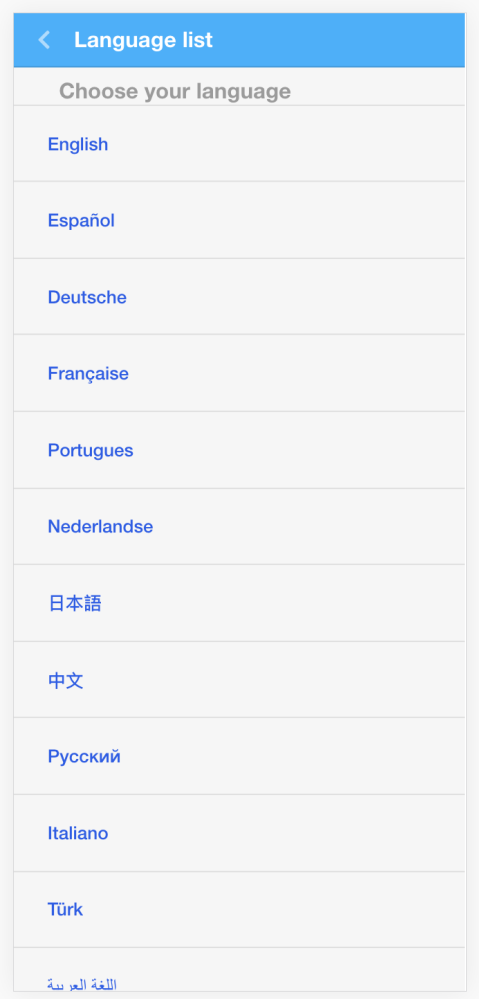
You can take a look at how it's implemented in **Demo App** in your account, screen - **13 Operation Loop Screen**.

⚠ You can take a look at how it's actually implemented in **Demo App** in your account, screen - **Operation Loop Screen**.

DATA FROM LISTS OF TABLES

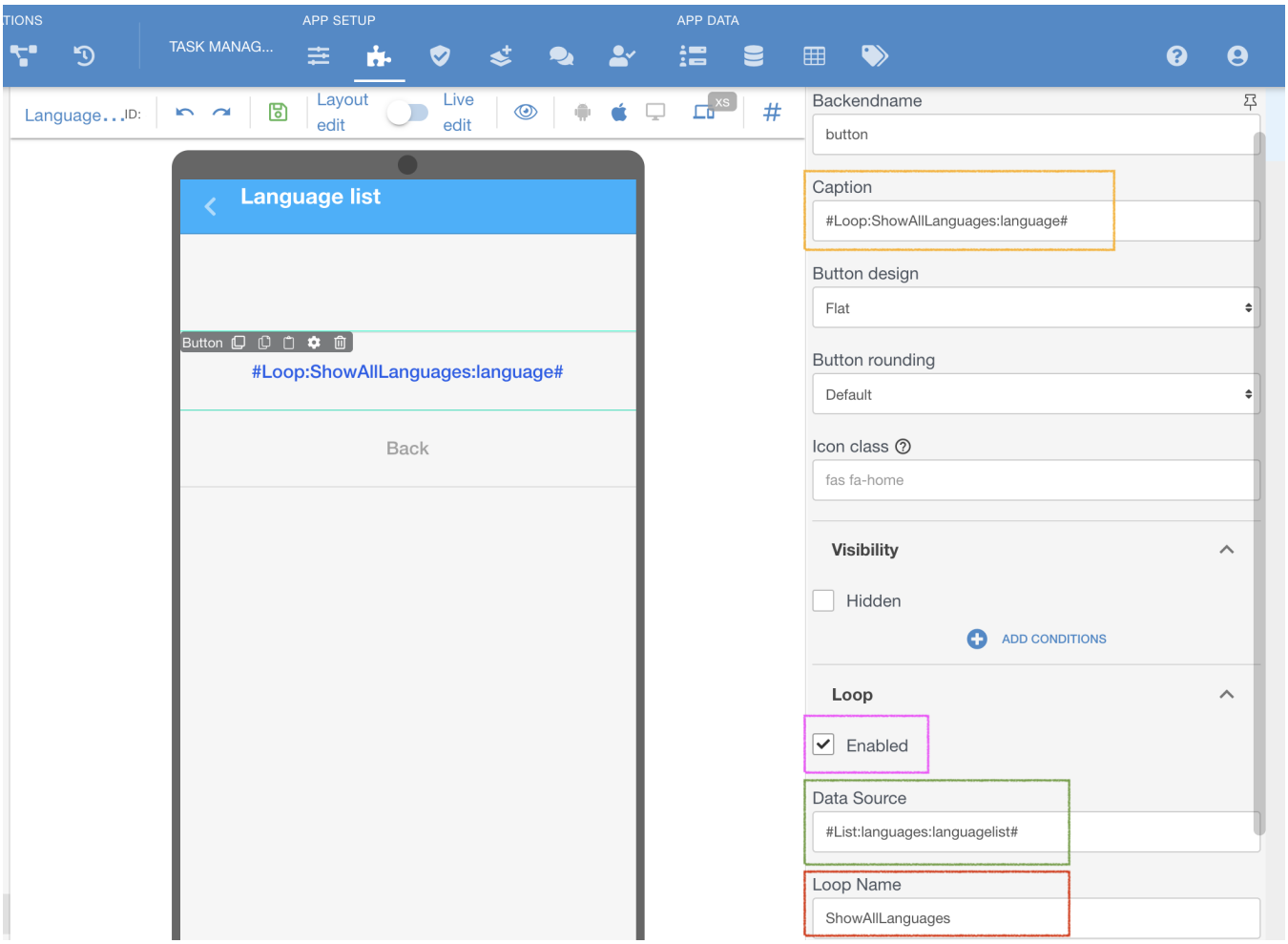
You can loop any data from any of your custom Lists of Tables. Pulling data arrays from lists of tables is available anywhere, where there is a Loop, including on slider components, etc etc.

For example, using existing tables in you Task Manager sample app, we can do this to show all languages available for it's users, which are stored in table named "languagelist" and in column named "language"



Steps taken in this example:

1. A button added to a screen
2. Enable **Loop** on the button, marked purple
3. Insert a hashtag with the List you want to use as Data Source, marked green , as example, **#List:languages:languagelist#**, where:
    a. **List** = a command to address lists of tables in hashtag
    b. **languages** = a name of a list we have in this sample app
    c. **languagelist** = a name of the actual table you are referring to
4. Name the Loop, as how it can be referred later on, marked red , for example - ShowAllLanguages
5. Refer to this named Loop in any place of a button, in this case in Caption section - #Loop:ShowAllLanguages:language# , where:
    a. **Loop** = a command to show that some data needs to be pulled from the Loop
    b. **ShowAllLangues** = a name of the loop we have given before
    c. **language** = an exact column name, which you are pulling from the list and need to iterate.



There is a reason why there is an extra step, from DECLARING the Loop's source and using it's data on screen in form of a Loop's name. This allows to declare and name more then one data source for the same iteration (inserting row in row and declaring loop on each of them) and refer to the exact needed data point later to show more complex data.

- App users - Objects
- Events of Users-Objects
- TroubleShooting the Wrapper
- General Intros
- Learn in Demo App