**3-5–App & Macro Customization** 

Apps and macros can make your workflows efficient but at the same time, they can complicate things when used incorrectly. In order to fully utilize the potential of these tools, you need to have a solid grasp of each type of macro, know your goals when converting a workflow to an app, and show your focus on optimal end-user experience when designing the interface as well. For this lesson, we are going to list some tips that can help you when developing your apps and macros, and as demonstrate the best steps to take to make your process dynamic.

**Arrange your questions strategically.** If you have several “questions” or interface tools that takes in user input, it is better to group related items together on each and prioritize the important questions on top. **The interface designer allows you to create tabs, groups, label, and move questions to organize your App UI.** To show the interface designer, navigate to the menu ribbon and click “View”. Make sure that the Interface Designer is checked. You can then pin or float this window to a portion of the designer window. The first tab of the interface designer is the Layout View. This shows the UI preview of your app or macro. In this example, we have an app which has 4 questions on it. You can move the questions by selecting one and using the arrow buttons. For this app, lets group the 2 numeric up down questions under “Model Data Allocation”. You can create a group by clicking the Add button, and selecting “Group Box”. You can change the group name by going to the configurations window and typing a new name on the Label Group text box. To push the questions inside the box, use the up or down arrow buttons until it is inside the box outline of the group. Next, lets add a label. Click the Add button again, and select “Label”. You can use the label to add additional instructions or information. To edit the label, navigate to the configuration window and enter the text on the label’s textbox. Type in “Test and Train Percentage must be <=100% in total.”. Once done, go back to the Layout view of the Interface designer window and place the label to the last part of the Model Data Allocation group.

In addition to editing the question placement, you can also use the Check box tool to create **detours in the app/macro**. On the last part of the app workflow, we have an output data tool which creates an yxmd file. We would like to make this optional so the end user can choose if they want an Output file or not. In order to do so, first place the output data tool into a container. Click right on the tool and select “add to new container”. Next, insert a Check box tool and place it above the created container. Let’s configure the Check box first. For its text, type in “Create an Output File.” e action type to “Enable/Disable Container from Question”. This way, the container will be disabled if the checkbox is not ticked. Since this output is the end process, make sure to put this checkbox on the bottom or last part of the app’s layout.

If you are having a hard time figuring out which question is under which group when looking at the layout view, another option for adjusting the placement and grouping of each question is via tree view. The tree view is the 3rd tab of the interface designer which shows the hierarchy of each question its underlying action. If you select one of the questions, it highlights which action is connected to it, as well as point you to its interface tool on the designer canvas. To move a question placement in the hierarchy, simply select the question and use the up or down arrows.

**Test the limits.** Make full use of the interface designer by using the test view tab. Import your test data and fill out the values in the test tab. Once done, click the button “Open Debug”. This creates a “debug workflow” on a new tab which applies the values that you’ve inputted in the test view. Notice that the interface tools are removed, so this works as a regular alteryx workflow. Another thing to take note of, is that if workflow tools reference connection names in an expression, the app values won't update in the debug workflow. Run this debug workflow to check if your app will generate errors based on the test data. You can then make the appropriate changes in the original app based on the errors. A debug log container is also available on the upper right portion of the debug workflow. This lists the app or macro values that you’ve used for testing as well as the action log of the debug run.

Aside the debug workflow, the test view also has the Reset button which lets you clear all entered values and return the defaults. Using the Save button, you can save your test values in an YXWV file to use in subsequent runs of the same Analytic App. This is extremely useful if you had to test a bunch of values to capture different results. You can then use the Open button to navigate to the saved YXWV file to re-use the values. Finally, the Open button will show the XML result of all questions with the values you've entered.

**Keep it clean.** Your once simple workflow can easily turn into a sea of webs and lines once you add a bunch of interface tools and action tools. To keep your back end clean and clutter free, make full use of the wireless connections. The best practice is to group the interface tools on the upper region of the workflow, then make the connection from action tool to individual tools as wireless. Simply click right on a line and select “Make Connection wireless” to hide it. You can still trace wireless connection by selecting the action tool to highlight the hidden line.

Another method to minimize connections is to employ Question Constants. **Question Constants let you skip the action tool and simply call the Interface Tool name to use its value.** For example, in this app, we have a formula tool which tags the Insurance Company name in the dataset. Its dummy data is set to “AVD Life”. As a first step in converting this portion to a user input value, we need to add the necessary interface tool to make it show up on the app interface. Drag a text box tool and place it above the formula tool. For the question to be displayed, type “Insurance Label:”. Then type “AVD Life” as the default value. Next, let’s change the tool name by navigating to the annotation tab and renaming it to “InsuranceLabel”. Since alteryx automatically list all questions as constants, you can navigate to Workflow Configurations window to check if the name is included as a Question Constant. Next, is to call the constant in the formula. Open the formula’s configuration and remove the “AVD Life” tag. Type in “'%Question.InsuranceLabel%'” to replace it with the constants value. Running it as an application will make alteryx call the question constant and use it as value instead of identifying the string as a text tag. But this method has its downsides. **Question Constants values cant be used in the debug mode, so any formulas or configuration that calls this constant will simply be regarded as strings in the debug workflow.**

**Place your macros in 1 folder.** To create easy access to all of the custom macros that you’ve built, place or save them to a single folder on your pc. Here we have the macros saved under a folder named “My Alteryx Macros” on the documents folder. You can then set this path as a search path for macros to display them on the tool palette. On alteryx designer, navigate to Options > User Settings > Edit User Setting. Then select the tab for “Macros”. Click the + button to add a new search path. Let’s set the category name as “My Macros” then paste the macro path in the search path or use the 3 dots button to navigate to it. Once done, hit Ok then save to close the user settings window. The “My Macro” category should now be available on alteryx designer as the last tool palette. You can simply drag and drop macros from this tab to use them in your workflow.