7. PAL UI Editor Functional Tests

* 1. Application-Level Operations
     1. *Opening*
        1. User launches the Editor
           1. Application window should open to a default resolution of 1200x800
           2. A single ‘New Procedure’ tab should be open upon startup
     2. *Resizing*
        1. User maximizes application to a resolution greater than the default
           1. Library & Variable Panel should remain the same size. Procedure view panel should grow
        2. User manually resizes window to a resolution smaller than 1200x800
           1. Is user prevented from resizing the window to less than the minimum resolution of 1000x500?
     3. *Closing*
        1. User closes the application (with no pending edits)
           1. Insure JVM associated with editor exits completely
           2. If PAL was running before the editor was launched, it should still be running after editor is closed
        2. User attempts to close the application with pending edits in any tab
           1. Is the user notified that he/she has pending edits and asked to confirm their desire to exit the application? (TLEARN-142)
        3. Is the user able to open another instance of the editor and reconnect to PAL successfully
     4. *Library*
        1. User adjusts the width of the library by dragging the pane-splitter
           1. Library should grow/shrink accordingly
           2. As the library grows, scrollbars should appear on the currently open procedure tab
        2. User shows/hides the library using the button on the toolbar
           1. Procedure view pane should consume the space created by hiding the library
           2. The hide/show button should respect any manual adjustments the user has made to the library’s width (remember last pane-splitter position)
        3. User has no action models loaded
           1. Application should still launch properly, library should appear blank with the exception of the toolkit
        4. User has multiple action models loaded
           1. Insure “Procedures” section lists all procedures across all namespaces
           2. Insure “Applications” section lists all actions from all namespaces
           3. Insure that each namespace has all of its actions listed and no actions from other namespaces
        5. User adds steps from the library to a procedure
           1. Insure that the “Recently Used” list is maintained as you add steps to the procedure
        6. An action model is loaded that has a namespace-level icon specification
           1. Any action that does not have its own icon specified in the action model should use the namespace-level icon in the library, info. Panel and on any instances of that step
        7. User has an action model loaded that specifies a special icon for a step
           1. The special icon should appear for that action when it appears in the library, info. Panel and on any instances of that step in a procedure
     5. *Tabs*
        1. User opens so many tabs that they can’t all be displayed in the tab header at once
           1. User should be provided with a menu that allows them to see all active tabs in a list and select any one of those tabs to become the active tab
        2. User closes all tabs
           1. Application should maintain a reasonable display.
           2. Subsequent openings of tabs (via Open or New) should work properly and reintroduce the tab pane.
  2. Procedure-Level Operations
     1. *Open*
        1. User opens a new (blank) procedure
           1. A new tab containing the blank procedure should appear
           2. Variable panel should be empty
           3. Application window title should read “New Procedure”
           4. If application window title previously indicated unsaved changes via an asterisk, that asterisk should now be gone.
           5. Undo/Redo buttons should be disabled
        2. User opens an existing procedure using the “Open” button on the toolbar
           1. The existing procedure should open in its own tab
           2. The tab’s Variable Panel should be populated with variables from the chosen procedure
           3. Application window title should be the name of the opened procedure
           4. Does the asterisk that indicates unsaved changes disappear?
           5. Is the undo stack cleared (undo/redo should become inactive)
           6. Do the variable panel and procedure view pane scroll bars adjust to the new data sets?
        3. User opens multiple existing procedures in multiple tabs
           1. Each tab should maintain its own undo/redo operations
           2. Each procedure tab should have its own Variable Panel. Tabs should never be able to influence the contents of each other’s Variable Panels
           3. When a tab is selected, its title should be reflected in the application window title
           4. Each tab should track the presence of unsaved changes separately and relay them (through an asterisk in the tab title) separately.
     2. *Save*
        1. User saves a procedure that has unsaved changes
           1. The asterisk in the window and tab title should disappear upon a successful save
           2. The undo/redo operations should *not* be cleared as a result of save
           3. Undoing the last edit should cause the asterisk to reappear in both the tab and window title
        2. User attempts to save a procedure that contains validation errors
           1. User should be prevented from saving until the errors are addressed
        3. User saves a procedure with no unsaved changes
           1. This is allowed and should work but is basically a no-op.
        4. User saves a blank/empty/new procedure using the Save button
           1. System should detect that the procedure is new and treat this operation as a “Save As” (same assertions as 7.2.3.1)
        5. User makes a “breaking change” to a procedure that is used as a step in another procedure. Breaking changes include anything that affects the number or type of procedure inputs/outputs.
           1. Is user informed that they cannot save such a change to a procedure that is used as a step and then allowed to save a copy?
     3. *Save As*
        1. User saves a copy of a procedure using the “Save As” button on the toolbar
           1. Is user prevented from saving the procedure with a name that is already in use?
           2. Is user prevented from including illegal characters in procedure name?
           3. Is user limited to 80 characters in for the new procedure name?
           4. The copy that is saved should become the active procedure in the tab
           5. The new copy of the procedure should appear immediately in the library.
           6. The copy should appear in subsequent openings of the procedure manager (either by performing another Save As or an Open)
           7. Undo/Redo should not be cleared as a result of a Save As, although there won’t necessarily be any undo items to begin with
           8. If there were pending edits at the time of the Save As button press then the asterisk should disappear after a successful Save As
        2. User presses “Save As” on a blank/new procedure
           1. Should work, same assertions as above
        3. User initiates “Save As” on a procedure with validation errors
           1. User should be prevented from saving until the errors are addressed
     4. *Rename*
        1. User attempts to rename a procedure using link in procedure header
           1. Is user forced to change the name or exit the window?
           2. Is user prevented from using a name that is already in use?
           3. Is the user prevented from entering illegal characters in the name?
           4. Is the rename reflected immediately in the procedure header?
           5. Is the rename reflected immediately in the tab title?
           6. Is the rename reflected immediately in the application window?
           7. Is the rename reflected immediately in the procedure header?
           8. Is the rename reflected immediately in the library & info panel?
           9. Is the rename reflected immediately in the “recently used” part of the library?
           10. Is rename link hidden from user on a new procedure?
        2. User attempts to rename the procedure that is open in the currently selected tab using the procedure manager (accessible by pressing the “Open” button and then using the “Actions” button)
           1. Same assertions as last test case
        3. User attempts to rename a procedure that is open but not in the currently selected tab using the procedure manager (accessible by pressing the “Open” button and then using the “Actions” button)
           1. Same assertions as last test case, except window title should not change
        4. User attempts to rename a procedure that is not open in a tab using the procedure manager
           1. Is the rename reflected immediately in the library?
           2. Is the rename reflected immediately in the “recently used” part of the library?
        5. User attempts to rename a procedure with unsaved changes (but no validation errors) from the procedure header
           1. Is user forced to save changes before proceeding with rename operation?
        6. User attempts to rename a procedure with unsaved changes (but no validation errors) from the Procedure Manager
           1. Is user forced to save changes before proceeding with rename operation?
        7. User attempts to rename an open procedure that contains validation errors using the link in the procedure header
           1. Is the user prevented from renaming the procedure and given a suitable explanation?
        8. User attempts to rename an open procedure that contains validation errors using the Procedure Manager
           1. Is the user prevented from renaming the procedure and given a suitable explanation?
        9. User renames a procedure that is referenced as a step in another procedure that is not opened in a tab
           1. Does the rename operation rename the reference in the other procedure? (open it to check)
        10. User renames a procedure that is referenced as a step in another procedure that is open
            1. Is the rename operation immediately reflected in the step of the other procedure?
            2. The rename should not affect the undo stack or unsaved changes status of the other procedure.
        11. User renames a procedure that is used as a step in a procedure that is open and has unsaved changes
            1. If the user abandons his changes (does not save), does the procedure reflect the rename next time it’s opened?
            2. If the user saves his changes, is the rename properly saved?
     5. *Close*
        1. User attempts to close a tab that contains a procedure with unsaved changes
           1. User should be prompted that they are about to throw away unsaved changes and asked to confirm
           2. If the user confirms, the tab should closed and be replaced by the tab that was immediately to its left (if such a tab exists)
     6. *Delete*
        1. User deletes a procedure that is not currently open (using the “Actions” button in the procedure manager, available via the “Open” button)
           1. Insure change persisted (procedure still gone after application is restarted)
           2. Procedure should immediately disappear from library, including the recently used list
           3. Procedure should not appear in any subsequent openings of the procedure manager
        2. User deletes a procedure that is currently open from the Procedure Manager
           1. Same assertions as above
           2. The tab for the deleted procedure should close automatically
        3. User deletes the procedure that is open in the currently selected tab
           1. Same assertions as last test case
        4. User attempts to delete a procedure that is referenced as a step in another procedure that is not opened in a tab
           1. Is user prevented from deleting the procedure and given a suitable explanation of why?
        5. User attempts to delete a procedure that is referenced as a step in another procedure that is opened in a tab
           1. Is user prevented from deleting the procedure and given a suitable explanation of why?
        6. User attempts to delete a procedure that is referenced as a step by a *newly added and unsaved* step in another procedure. It’s important that the newly added step be the only reference to the procedure as a step in order for this to be a good step.
           1. Is user prevented from deleting the procedure and given a suitable explanation of why?
     7. *Import/Export*
        1. User exports a basic procedure
           1. Does export succeed with no errors? File created on disk?
           2. Upon performing a subsequent export, does the file browser window remember its location?
           3. Is the procedure saved with a .procedure extension? It should be impossible to save with any other extension
        2. User exports a procedure that references another procedure
           1. Export should result in the creation of a single file that references but does not define the other procedure
        3. User imports a basic procedure using the Open button
           1. Does the import succeed with no errors?
           2. Does the file browser window remember its location?
           3. Library should immediately reflect the presence of the new procedure. Same for the procedure manager.
        4. User imports a procedure that references another procedure as a step that is defined/loaded in the system and has a matching signature
           1. Same assertions as above
        5. User imports a procedure that references another procedure as a step that is defined/loaded in the system and does not match its signature
           1. User should be prevented from importing that procedure
        6. User imports a procedure with no inputs (TLEARN-367)
           1. Insure no exception is thrown during input
        7. User imports a procedure with a name that is already in the system
           1. Is the user notified that they are about to overwrite a procedure?
           2. Is the existing procedure successfully overwritten? (Introduce a change in the existing procedure, save your changes and then verify that the change is overwritten upon import)
        8. User imports a procedure with a name that is already in the system. The procedure already in the system is referenced as a step in another procedure.
           1. Is the user notified that they cannot overwrite this procedure because it is used as a step in another procedure?
           2. Is user allowed to continue the import under a different name?
        9. User imports a procedure that uses outdated action model
           1. TODO: Define correct behavior
     8. *Edit Description*
        1. User adds a description to a procedure
           1. Is user limited to 400 characters when modifying a procedure’s description?
           2. Is an undo action added for the description change?
           3. Does application window title reflect that the procedure has been modified with an asterisk?
           4. Can the action be undone/redone successfully?
           5. Is the procedure marked as having unsaved changes if it did not previously?
           6. Upon save, is the change persisted?
        2. User deletes (blanks out) a procedure description
           1. Same assertions as 2.8.1
        3. User modifies an existing procedure description
           1. Same assertions as 2.8.1
        4. User adds a very long description
           1. Same assertions as 2.8.1
     9. *Execution*

*Note: Testing execution will require an instrumented test application, such as Novo*

* + - 1. User attempts to run a procedure with no inputs by clicking the “Run” button
         1. Procedure should run immediately
         2. Is procedure button disabled while procedure is still running?
      2. User attempts to run a procedure with inputs
         1. Does a modal window appear that allows the user to specify input values?
         2. Does the ‘run’ window warn users about null/empty input values?
         3. Try specifying different input values across different types and make sure they actually impact the procedure execution
      3. User attempts to run a procedure with unsaved changes
         1. Is user forced to save their changes before proceeding with execution?
      4. User attempts to run a procedure with validation errors
         1. Is the user prevented from doing this in some way?
    1. *Stepped Execution (“Debug” execution)*
       1. User debugs a procedure with no inputs
          1. Application should go straight in to debugging mode
          2. New buttons for stop, next and finish should appear on the toolbar
          3. The tab pane should become locked, preventing the user from changing the active tab
       2. User debugs a procedure with inputs
          1. Same assertions as above except that the user should be prompted with a procedure input dialog first.
       3. User wants to inspect variable values during execution
          1. All procedure inputs and variables bound in a step prior to the step the procedure is currently paused on should include a “Debug Value” specification in their tooltip that lends insight in to the value represented by the variable
          2. The result of the current step should not have a “Debug Value” as we don’t know what its value is yet. The same goes for subsequent steps.
          3. If a variable that was bound by a previous step is referenced in a subsequent step, the subsequent reference should include the “Debug Value”
       4. User steps through a procedure that includes a loop
          1. Insure no exceptions as this code has been problematic in the past
          2. Insure iterand variable has a “debug value”
          3. The debug values for outputs of steps contained within a loop should reset each time through the loop (on 2nd time through the loop, insure that no variables scoped within the loop contain debug values as those values are no longer valid).
          4. It should not be possible to inspect the “debug value” of an output from the last step in a loop
       5. User steps through a procedure that includes a loop with an accumulator
          1. Accumulator should have a debug value after the loop step is completely finished
       6. User steps through a procedure that includes a parallel iteration loop
          1. Make sure everything works properly and that the iterands have debug values in their tooltip.
       7. User finishes debugging a procedure (runs out of steps)
          1. Debug buttons should disappear
          2. Tab pane should become active again
          3. Run/Debug button should be enabled again
       8. User chooses to “Finish” debugging a basic procedure
          1. Procedure should finish executing automatically
          2. Same assertions as above
    2. *Configuring Default Values for Procedure Inputs*
       1. User attempts to modify default value of a procedure input
          1. Is user able to modify the default value either directly using the control to the right of the input button, or indirectly (through a button control) for structs and collections?
          2. Is an appropriate undo created? Can you undo/redo?
          3. Does the application recognize this as an unsaved change?
          4. Editing the default value of a procedure input should have no impact on references to that input elsewhere in the procedure
       2. User attempts to clear default value for a procedure input
          1. Is user able to clear default value using a menu option from the menu button representing the input? (“Ask the user for this value”)
          2. Is this recognized as an unsaved changed?
          3. Is an appropriate/functional undo/redo created?
       3. After clearing a default, can the user once again specify a default value by selecting to “Provide a default” from the input’s menu options?
          1. Does an appropriate editing control replace the disabled “Ask the user” textbox?
          2. After the user actually provides a new default value, is this recognized as an unsaved change? Undo/redo created/functional?
    3. *Reordering Procedure Inputs*
       1. User moves input down
          1. Up/down arrows should have tooltips
          2. Should have no impact on variable panel
          3. Up arrow on top-most input should be disabled
          4. Down arrow on bottom-most input should be disabled
          5. Should be able to save/persist change
          6. Was an undo action created? Can it be undone/redone?
          7. Is the procedure marked as having unsaved changes if it did not previously?
          8. Upon save, is the change persisted?
       2. User moves input up
          1. Same assertions as above
       3. User moves top input down
          1. Same assertions as above
       4. User moves bottom input up
          1. Same assertions as above
       5. User moves top input to bottom
          1. Same assertions as above
       6. User moves bottom input to top
          1. Same assertions as above
    4. *Publishing Results (procedure outputs)*
       1. User publishes a result on a procedure with no existing published results
          1. None of the variables in the publishing dialog should have their check-box checked when dialog is first opened.
          2. User should be able to freely check or uncheck multiple results
          3. Upon clicking “Ok” with at least one variable checked, the dialog should disappear and the procedure footer should visualize each published variable in alphabetical order
          4. Is an undo action created? Can it be undone/redone?
          5. Is the procedure marked as having unsaved changes?
          6. Change should be persisted upon save.
       2. User removes all published results from a procedure
          1. Procedure footer should clear all published results from the procedure footer
          2. Is an undo action created? Can it be undone/redone?
          3. Is the procedure marked as having unsaved changes?
          4. Change should be persisted upon save.
       3. User publishes all step outputs
          1. Is an undo action created? Can it be undone/redone?
          2. Is the procedure marked as having unsaved changes?
          3. Change should be persisted upon save.
          4. Change should be persisted upon save.
       4. User uses a procedure with published results as a step within another procedure
          1. All published results should appear as step outputs. (Use procedure modified in 7.2.12.3)
       5. Ensure user is prevented from publishing loop iterands or variables that are bound by steps contained within loops (TLEARN-474)
    5. *Error/Warning Navigation*
       1. User has introduced only warnings in to the procedure
          1. A button should appear on the toolbar with the warning icon
          2. Clicking on the button should result in the procedure view pane scrolling to each warning in turn and selecting the step that contains the warning.
          3. After clicking through all warnings, another click should result in cycling back to the top and finding the first warning.
       2. User has introduced errors (and possibly warnings) in to the procedure
          1. A button should appear on the toolbar with the error icon
          2. Clicking on the button should result in the procedure view pane scrolling to each warning/error in turn and selecting the step that contains the error/warning.
          3. After clicking through all errors/warnings, another click should result in cycling back to the top and finding the first error/warning
       3. User has introduced errors to a procedure while other procedures are open in other tabs
          1. Insure that the error button is only visible when the tab that contains the procedure with errors is selected
    6. *Visualization*
       1. Tester should open each canned procedure and visually inspect how its rendered in the procedure view pane. Look for inappropriate wrapping, overrunning of visual bounds, etc.
  1. Step-Level Operations
     1. *Step Selection*
        1. User selects steps using the selection rectangle
           1. User should be able to start a selection rectangle by clicking and dragging anywhere in the procedure view pane except on the body of a step
           2. A selection rectangle should not be created when you click on the body of a step (this should instead trigger step movement behavior)
           3. Should be able to begin a selection rectangle by clicking in the “whitespace” of a loop step. In other words, the area within the loop that is not occupied by a sub-step.
           4. Dragging the rectangle over a step should cause it become selected
           5. Mutating the rectangle such that it no longer covers a step should cause that step to become unselected
           6. It should not be possible to select two steps that do not share the same parent
           7. It should not be possible to select a loop as well as a step it contains (directly or indirectly)
           8. It should not be possible to select two steps contained within different loops
           9. It should not be possible to select a step that is outside of a loop and a step that is within a loop.
           10. It *should* be possible to select two steps directly inside of the same loop, or two steps that are both top-level (not within a loop)
        2. User selects a single step with a mouse click
           1. Clicking on the body of a step should cause that step to become selected
           2. Clicking on the body of another step without holding the control or shift key should cause the currently selected step to become unselected and the clicked step to become selected
        3. User selects multiple steps using control keys & mouse clicks
           1. Holding the shift or control key and clicking on a step should cause it to be added to set of currently selected steps without unselecting any steps
           2. Holding the shift or control key and clicking a step that is already selected should cause it to become unselected
           3. It should be possible to release the control/shift key between step selections, provided that the user is pressing control/shift at the time of the next step click
           4. It should be possible to select non-contiguous steps in this manner
        4. User clicks off of a selection
           1. User should be able clear any current selection by clicking in the whitespace of the procedure view pane while not holding the shift or control key
     2. *Step Deletion*
        1. User deletes single step using delete button
           1. Does the step disappear immediately upon pressing the delete button?
           2. Are all non-selected steps retained?
           3. Are the steps numbered correctly after the deletion
           4. Was an appropriate undo action created?
           5. Can the action be undone/redone successfully?
           6. No errors upon save
           7. Is the deletion persisted upon save?
        2. User deletes multiple steps using delete button
           1. Same assertions as 3.2.1 but with multiple steps
        3. User deletes single step by dragging it off of the procedure view pane
           1. Same assertions as 3.2.1 but with multiple steps
        4. User deletes multiple steps by dragging them off of the procedure view pane
           1. Same assertions as 3.2.1 but with multiple steps
        5. User deletes a loop
           1. Are the loop and all steps it contained (directly or indirectly) deleted?
           2. Same assertions as 3.2.1
        6. User deletes last step in a loop
           1. The loop should remain and still render correctly
           2. It should be possible to add or move another step inside of the loop
        7. User deletes the last step in a procedure
           1. Does the UI still look good?
        8. User deletes last step that references a procedure input variable
           1. Does a warning icon appear next to that procedure input in the procedure header?
           2. Is it still possible to save/run the procedure without addressing the warning?
           3. Is the icon clickable and does it have the effect of removing that procedure input from the procedure?
           4. Is the variable panel on the right updated appropriately?
           5. If you don’t address the warning, does it reappear when the procedure is opened again in a separate editing session?
        9. User deletes an action that was responsible for binding a procedure output
           1. Is the user prompted that by deleting the step he/she will cause the procedure output to be deleted as well?
           2. Is the procedure output removed from the variable panel as well as the procedure output configuration dialog (accessible by clicking the link in the procedure footer)
           3. If you undo the step deletion, does the procedure output reappear?
        10. User deletes a step whose output was referenced as an input to subsequent steps
            1. Is an error reported to the user on the toolbar and in the procedure view?
            2. Is there an appropriate tooltip on the error icon on the step informing the user of the error and that clicking the error icon will allow them to fix error?
            3. Upon clicking the icon, the user should receive the following repair suggestions. Verify that each is present and fixes the problem upon being selected:

Undo (only present if there were no errors before the last error generated)

TODO Flesh out the repair operations

* + - 1. User opens a procedure that’s only reference to a procedure input is contained within a collection (list or function call).
         1. Verify that the user does *not* receive a warning on the procedure input indicating that it is unused.
         2. Use canned procedure 6.1NestedVarReference
    1. *Step Movement*
       1. User hovers mouse over the knurling of a step
          1. Does cursor change to a hand to indicate that a step can be dragged?
          2. Does an appropriate tooltip appear?
       2. User moves a single step by clicking and dragging on the knurling
          1. If the step was not previously highlighted/selected, does it become selected upon being dragged?
          2. Does the step look the same being drug as it did when it was in the procedure, with the exception of being less opaque?
          3. Step numbers should not change while the step is being dragged. Only after dropping.
          4. As the step is dragged over the procedure view pane, is space created to indicate where the step would be dropped if the user were to release the mouse button?
          5. Upon releasing the mouse, does the step appear in the appropriate location in the procedure? Does the procedure view refresh appropriately?
          6. Are the steps numbered correctly after the move?
          7. Was an appropriate undo action created?
          8. Can the action be undone/redone successfully?
          9. No errors upon save; Is change persisted upon save?
       3. User moves a single step by clicking and dragging somewhere else in the step header that is not occupied by a term, button etc
          1. Same assertions as 7.3.3.2 but the cursor will not change to a hand
       4. User attempts to move a step by clicking on a term or button contained within the header of the step they wish to move
          1. This should not work. Depending on what they click, it might open up a menu button
       5. User initiates a single step move but replaces the step in its original location
          1. Should not result in creation of an undo
          2. Should not result in the procedure being marked as having unsaved changes
          3. Are steps still numbered correctly?
       6. User drags a step over the procedure header and releases the mouse
          1. Upon being dropped, the step should become the first step in the procedure.
          2. Same assertions as 7.3.3.2.
       7. User drags a step over the procedure footer and releases the mouse
          1. Upon being dropped, step should become the last step in the procedure
          2. Same assertions as 7.3.3.2
       8. User moves multiple contiguous steps
          1. Upon dragging one step of a multi-step selection, all of the selected steps should be removed from the procedure view. They should be replaced with a single “step” that follows the drag across the application and contains a description of the steps it represents (multiple steps are represented as a single unit while being dragged)
          2. Once dropped, are the moved steps in the same relative order?
          3. Are all of the steps in the procedure numbered correctly after the drop?
          4. Was an appropriate undo action created?
          5. Can the action be undone/redone successfully?
          6. No errors upon save. Is the change persisted upon save?
       9. User initiates a multi-step move involving contiguous steps but replaces them in their starting location
          1. Same assertions as 7.3.3.5
       10. User moves multiple non-contiguous steps
           1. Same assertions as 7.3.3.8
           2. Steps should be contiguous in the same relative order after being dropped
       11. User initiates a multi-step move involving non-contiguous steps and replaces the drag step in its starting location
           1. Different case than 7.3.3.9. Since the steps were non-contiguous, they should now be contiguous in the drop location even though it is the starting location.
           2. Same assertions as 7.3.3.10
       12. User moves single step into a loop
           1. Same assertions as 7.3.3.2
       13. User moves single step out of a loop
           1. Same assertions as 7.3.3.2
       14. User moves multiple contiguous steps in to a loop
           1. Same assertions as 7.3.3.8
       15. User moves multiple non-contiguous steps in to a loop
           1. Same assertions as 7.3.3.10
       16. User moves multiple contiguous steps out of a loop
           1. Same assertions as 7.3.3.8
       17. User moves multiple non-contiguous steps out of a loop
           1. Same assertions as 7.3.3.10
       18. User moves a loop step
           1. The body of the loop should collapse while being dragged, such that the sub-steps of the loop are not visible
           2. Same assertions as 7.3.3.2
       19. User moves a loop step and a non-loop step
           1. Should be treated like any other multi-step move
           2. Same assertions 7.3.3.8 or 7.3.3.10
       20. User moves two loop steps
           1. Should be treated like any other multi-step move
           2. Same assertions 7.3.3.8 or 7.3.3.10
       21. User moves a step such that it reference a variables before it is declared/bound
           1. Is an error reported to the user on the toolbar and in the procedure view?
           2. Is there an appropriate tooltip on the error icon on the step informing the user of the error and that clicking the error icon will allow them to fix error?
           3. Upon clicking the icon, the user should receive the following repair suggestions. Verify that each is present and fixes the problem upon being selected:

Undo (only present if there were no errors before the last error generated)

TODO Flesh out the repair operations

* + - 1. User moves a step such that it references a variable that is no longer in scope (a loop iteration variable)
         1. Same assertions as 7.3.3.21
    1. *Step Addition*
       1. User adds an action with no inputs from the library
          1. Is a space created beneath the step as it’s dragged over the procedure view to indicate the current drop location?
          2. Can the user scroll the procedure view with the mouse wheel while dragging to find the right drop location?
          3. Is the step dropped without a dialog appearing to configure inputs since there are none?
          4. Are the steps numbered appropriately after the drop?
          5. Is an undo action added?
          6. Can the action be undone/redone successfully?
          7. Can the change be persisted?
          8. Is the action added to the “Recently Used” list?
       2. User adds an action with inputs
          1. Is the user prompted with a dialog to configure the inputs to the step?
          2. Can the user successfully cancel the step addition from the dialog?
          3. Is the user required to configure each input before being allowed to proceed?
          4. Check each of the following assertions for each input required by the step:
          5. Is the user provided with the correct list of existing variables that can be used?

Are variables that are defined after the drop location excluded from the suggested variables?

Are variables of a type that is disparate from the type of the action input excluded from the suggested variables?

* + - * 1. Is the user offered to provide a constant value?
        2. Is the user provided with the correct constant editor if they select to provide a constant
        3. Can user create a procedure input for the variable’s value?
        4. Upon clicking “Okay”

Is an undo action added?

Can the action be undone/redone successfully?

If a procedure input was created for one of the action’s inputs, is it also removed as part of the undo?

Can the change be persisted?

Is the action added to the “Recently Used” list?

* + - 1. User adds an action with outputs
         1. Is a unique/appropriate variable name generated for the output variable?
         2. Does each output appear in the variable panel on the right?
         3. Do the new variables appear in the suggested replacements for terms that exist below the drop location (and are in scope?)
      2. User adds step to an empty (new procedure)
         1. Behavior should be the same as when adding to an existing procedure.
         2. Insure that the procedure view renders correctly
      3. User adds an action from a different namespace in the library
         1. Same assertions/behavior as 7.3.4.1/7.3.4.2/7.3.4.4 depending on the step
      4. User adds a procedure step
         1. Should work the same as normal actions
         2. Only difference visible to the user is that they select the step from the “Procedures” portion of the library
         3. Same assertions/behavior as 7.3.4.1
      5. User adds a procedure with inputs as a step
         1. Same assertions/behavior as 7.3.4.2
      6. User adds a procedure with published results as a step
         1. Same assertions/behavior as 7.3.4.3
      7. User attempts to add the current procedure as a step to the current procedure
         1. User should be prevented from doing this because they will create an infinite loop
      8. User attempts to add a procedure that relies on the current procedure (directly or indirectly) as a step to the current procedure
         1. User should be prevented from doing this because they will create an infinite loop
      9. User adds a loop when there is an existing variable suitable for iteration
         1. User should be prompted with a dialog to configure the loop
         2. Should be able to select existing collection variables as the collection to be iterated over
         3. User should be prevented from proceeding until the loop is properly configured
         4. Upon clicking okay, the system should have generated an iteration variable of the appropriate type given the type of the collection.
         5. Check for appropriate tooltips on both the input collection and iteration variable in the loop header and footer
      10. User adds a loop where this not an existing variable suitable for iteration
          1. Since we currently do not support adding inline function calls/collection generation, if there are no variables that are suitable for iteration then it is impossible to configure the loop properly
          2. TODO define correct behavior (at least for WebTAS 4.3)
      11. User adds an action from the “Recently Used” part of the library
          1. Should work just as if it came from its normal location(s) in the library
      12. User drags step from the library and drops it on the procedure header
          1. Should act as if the step was dropped in the first step location of the procedure
          2. Same assertions as for other step additions
      13. User drags step from the library and drops it on the procedure footer
          1. Should act as if the step was dropped in the last step location of the procedure
          2. Same assertions as for other step additions
      14. User drags step from the library but drops it outside the bounds of the procedure view pane
          1. The step the user was attempting to add should simply disappear and this becomes a no-op
          2. No undo, no “unsaved changes” indicator, etc.
    1. *Existing Loop Configuration*
       1. Do all loop steps provide a wrench icon with an appropriate tooltip that allows for configuration of the loop?
       2. User reconfigures a loop to iterate over a different collection of the same type
          1. Upon clicking the wrench, x configuration dialog should appear that is exactly the same as the dialog that appears when the user drags a loop from the library’s toolbox
          2. Dialog should include a menu button that includes all legal replacements for the collection that is currently being looped over
          3. User should be able to cancel out of the loop configuration
          4. Upon choosing a replacement that is of the same type and clicking Okay on the dialog, the dialog should disappear and the loop should now be set up to loop over the new collection.
          5. The name and type of the iteration variable should not change
       3. User reconfigures a loop to iterate over a different collection with a different collection type (Eg list -> set or set -> bag)
          1. Same assertions as 7.3.5.2
       4. User reconfigures a loop to iterate over a collection with a different element type
          1. Previous iterand should disappear and be replaced by a new iterand of appropriate type. This will almost certainly cause validation errors.
          2. Does undo/redo work appropriately?
    2. *Step details expansion*
       1. Does each step in the procedure have a button (in the upper right) that allows the user to toggle the step to display more details?
       2. User expands details of a step with inputs and outputs
          1. Details should include the name and description of the step from the action model
       3. User expands details of a loop step
          1. Details should include a explanation of a loop
       4. User expands details of a procedure step
          1. Should include the name and description of the procedure.
  1. Term-Level Operations
     1. *Term Inspection*
        1. User explores the nature of the terms contained within a procedure
           1. All types of terms (procedure inputs, variables, constants, functions, lists, null, etc) should have explanatory tooltips that include information as to their nature and type.
           2. The tooltips should be available anywhere the term appears inside of the procedure view pane. The tooltip of variables and procedure inputs contained within the variable panel on the right will differ
     2. *Term replacement (not using copy/paste)*
        + 1. *Assertions: each of the following test cases should result in the creation of an undo. Test the ability to undo and redo each action.*
          2. *After performing several of these test cases in a row, go through the entire stack of undo operations and insure you can undo and redo everything. Some bugs will only show up after a combination of undo operations have taken place*
          3. *Note that all of these actions take place on the inputs to various steps within a procedure. “Existing value” refers to variables that were the output of a previous step.*
        1. User replaces a variable with a new procedure input (“Ask the user…”)
        2. User replaces a variable with an existing procedure input
        3. User replaces a variable with a constant
        4. User replaces a variable with another variable
        5. User replaces a variable with a function call
        6. User replaces a procedure input with another procedure input
        7. User replaces a procedure input with a new procedure input
        8. User replaces a procedure input with an existing value
        9. User replaces a procedure input with a constant
        10. User replaces a procedure input with a function call
        11. User replaces a constant with an existing procedure input
        12. User replaces a constant with a new procedure input
        13. User replaces a constant with an existing value
        14. User replaces a constant with a function call
        15. User replaces a function call with an existing procedure input
        16. User replaces a function call with a new existing procedure input
        17. User replaces a function call with an existing value
        18. User replaces a function call with a constant value
        19. User replaces a null value with a procedure input
        20. User replaces a null value with a constant
        21. User replaces a null value with an existing value
        22. User replaces a null value with a function call
     3. *Copying and Pasting of Terms*
        + 1. *TODO: define copy/paste functionality and write appropriate test cases*
     4. *Variables / Step (Action) Outputs*
        1. User expands the menu button of an action output
           1. The drop-down menu for an action output should never include options to use an existing value, procedure input or constant value.
           2. The menu should always include an option to rename the output variable and to highlight all occurrences
     5. *Variable Rename and Rename/Replaces*
        + 1. *Note: a Rename/Replace refers to a special operation that occurs when the user renames a variable and gives it the name of another variable of the same type in the procedure. The user can do this when only one or neither of the variable names in question are bound, but not when both are bound. To test this correctly, you need to understand the difference between a variable binding and a variable reference. References to an unbound variable should always be denoted by an error on the step. See TLEARN-496 for more details.*
        1. User renames a procedure input to an entirely brand new name
           1. Is the new name limited to 22 characters?
           2. Is an appropriate undo action created?
           3. Can the action be undone/redone?
           4. Try doing this both from the procedure header and from a reference to the input in the procedure body and variable panel
        2. User renames an “existing value” (step output) to an entirely brand new name
           1. Same assertions as above
        3. Use attempts to rename a variable to a value that only differs in capitalization when compared to another variable
           1. Unless one of the variables is unbound and both variables are of the same type, the user should not be able to perform this rename. Although the application tracks the capitalization of variable names, it prevents the user from creating two distinct variable names that only differ by capitalization.
        4. User renames a variable and only changes capitalization
           1. The system should allow this.
           2. The capitalization changes should take effect immediately in the visualization(s)
           3. An undo/redo should be created just like a normal rename
        5. User renames a variable that is not bound to the name of another variable of the same type that is bound
           1. The user should be allowed to do this
           2. Upon completion of the rename, select to “highlight all occurrences” of the surviving variable name and insure that *all* references to that variable name become highlighted, which insures that the two variables were merged completely.
           3. An undo/redo should be created and should work bidirectionally.
        6. User renames a variable that is bound to the name of another variable that is referenced but not bound of the same type
           1. Same assertions as above.
        7. User renames a variable that is not bound to the name of another variable that is referenced but not bound of the same type
           1. Same assertions as above
        8. User attempts to rename a variable that is bound to the name of another variable that is bound
           1. User should be prevented from doing this in the rename dialog because both variables are bound.
        9. User attempts to rename a variable that is not bound to the name of another variable of a *different* type
           1. User should be prevented from doing this because the two variables are not of the same type and therefore a rename/replace operation won’t work
        10. User attempts to rename a procedure input to have the same name as another procedure input
            1. User should always be prevented from doing this because procedure inputs are always bound, therefore this isn’t a legal rename/replace case.
        11. User adds a new step and chooses to “ask the user” for one of the step’s inputs. They then rename the new procedure input (in the add step dialog).
            1. Is the user allowed to rename the input to the name of an existing variable of the same type that is unbound?
            2. Is the user prevented from renaming the input to the name of any existing variable that doesn’t meet the above specifications including a variable of different type, a variable that is bound, etc?
        12. User adds a new step and chooses to “ask the user” for two inputs that are of the same type. The user renames both inputs (in the dialog) to the same variable name and hits “OK”
            1. Is the step added successfully?
            2. The application should have known to combine the two “ask the users” to a single procedure input.
            3. Insure undo/redo works correctly.
        13. Insure that the application considers loop iteration variables to be bound
        14. Insure that the application considers loop accumulators to be bound
        15. Insure that the application considers all step outputs to be bound
        16. Insure that the application considers all procedure inputs to be bound
        17. Rename/replace can cause a variable to be referenced before it is bound or referenced outside of its scope, just like moving steps can. Cause some of these errors to occur and insure the following:
            1. That the errors are immediately reported as usual
            2. That the errors can be resolved either through undo, deleting steps, etc
        18. User renames a loop iteration variable
            1. User should be able to rename it to an entirely new name, or the name of an existing v
            2. Insure this works correctly and that the visualizations update accordingly.
     6. *Constant Editing*
        1. User edits the value of an instance of our built-in string primitive type
        2. User edits the value of an instance of our built-in integer primitive type
        3. User edits the value of an instance of our built-in datetime primitive type
           1. **Incomplete**
        4. User edits the value of an instance of our built-in duration primitive type
           1. **Incomplete**
        5. User edits the value of an instance of our built-in Boolean primitive type
        6. User edits the value of an instance of a custom type that uses java.lang.String
        7. User edits the value of an instance of a custom type that uses java.lang.Integer
        8. User edits the value of an instance of a custom type that uses java.lang.Long
        9. User edits the value of an instance of a custom type that uses java.lang.Short
        10. User edits the value of an instance of a custom type that uses java.lang.Float
        11. User edits the value of an instance of a custom type that uses java.lang.Double
        12. User edits the value of an instance of a custom type that uses java.lang.Boolean
        13. User edits the value of an enum constant
        14. User edits the value of a struct constant
        15. User edits a collection of strings
        16. User edits a collection of enums (verify correct nested editor)
        17. User edits a collection of ints (verify correct nested editor)
        18. User edits a collection of doubles (verify correct nested editor)
        19. User edits the value of a collection nested within a collection
        20. User edits the value of a struct nested within a collection
        21. User edits the value of a collection nested within a struct
        22. User attempts to edit a procedure that references a custom type not on the editor’s classpath (ardaFoo)
        23. User attempts to replace a variable with a variable that is a subtype of the current variable
            1. Use canned procedure 2.7CustomTypeInheritance to make sure user can switch to variables they should be able to and prevented from switching others, according to the rules of inheritance
        24. User attempts to replace a variable with a variable of an “equivalent” type
            1. Use canned procedure 2.6EquivalentTypes and insure that you can swap out variables that have equivalent types, and that other types are excluded appropriately
     7. *Functions*
     8. *Types*
        1. User deletes a step that was binding an output variable. The variable was referenced as an input to subsequent steps. The **user** then attempts to add a new step with an output and bind the output to the missing variable (with the same name)
           1. The user should not be able to rename the output to the name of the unbound (missing) variable. Otherwise, we run the risk of allowing the user to create a type mismatch.
           2. To handle this case, the user must fix the references to the old name to point to a new/separate variable. Once all errors have been addressed, the user should be able to rename the variable to the original name.
        2. Same as 7.4.7.1 except that rather than adding a new step, the user attempts to repurpose the output of an existing step by renaming its output to the missing variable name.
        3. Insure that when the action model contains two custom types that share an underlying representation type (eg Java.lang.String) the editor does not allow the user to use these types interchangeably as they are semantically separate.
        4. Insure that when an action model utilizes struct inheritance that the editor will allow an instance of a super-struct to be replaced by instances of any of its sub-structs.
           1. **(System support for struct inheritance is incomplete)**
     9. *Highlighting*
        1. User highlights a procedure input by clicking it in the procedure view pane and selecting “Highlight all occurrences”
           1. Are all occurrences of the procedure input highlighted?
           2. Any previously highlighted terms should have become un-highlighted
        2. User highlights a variable by clicking it in the procedure view pane and selecting “Highlight all occurrences”
           1. Are all occurrences of the variable highlighted?
           2. Any previously highlighted terms should have become un-highlighted
        3. User highlights a procedure input from the variable panel
           1. Same assertions as 7.4.9.1
        4. User highlights a variable from the variable panel
           1. Same assertions as 7.4.9.2
  2. Miscellaneous