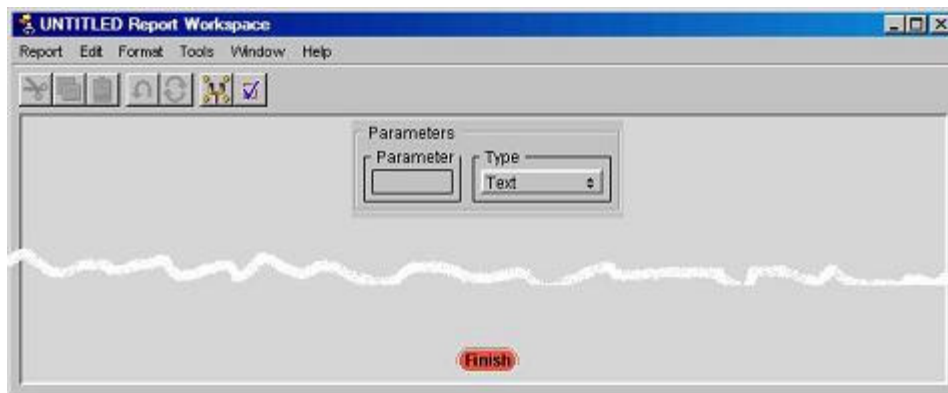


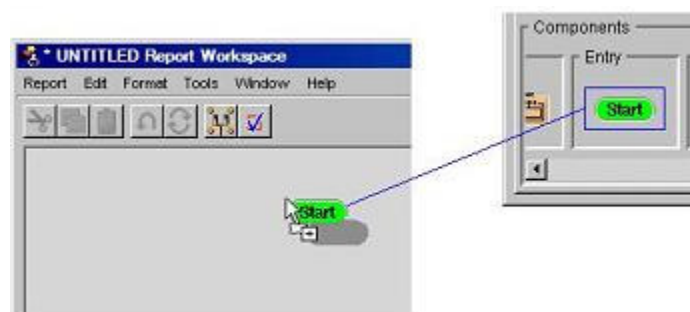
Tutorial 2 - Building the CustomerList Report Procedure

Report Procedures implement the logic of the report, and make up the Business Logic layer of Scribe. The CustomerList Report Procedure is quite simple in its design, yet it will help you learn the basic concepts of graphical programming of the executable flowcharts, and the operation of one of its most important components - Database Retrieval.

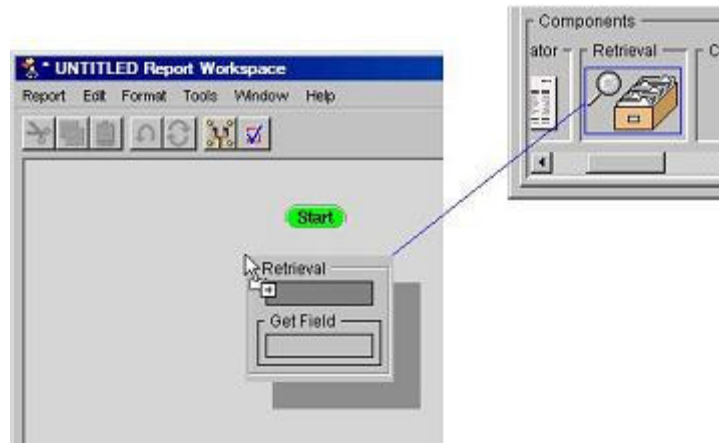
Select Report Procedures entry in the Application Launchpad window and click on the Launch button (or just double-click the Report Procedures entry). Scribe will open a new ("UNTITLED") Report Procedure Workspace window:



Initially, every new Report Procedure Workspace is loaded with the Parameters component, which is a vehicle for passing run-time input parameters to the procedure. However, the report we are assembling does not need any input parameters, so we will select this component by clicking on it (this selects the component and makes it a target of the next action), and delete it by pressing Delete or Backspace key. In its place, we will drag-and-drop a parameterless ("blind") Entry component from the bottom of the Palette window:



Now we need to organize a retrieval loop that will extract all required customer data from the Customers table, and pass it to the Print Template created in the Tutorial1. At the bottom of the Palette window there is a Retrieval component; it extracts data from the database table and passes it to the procedure. Drag the Retrieval component into the Workspace and drop it at the location shown in the figure:

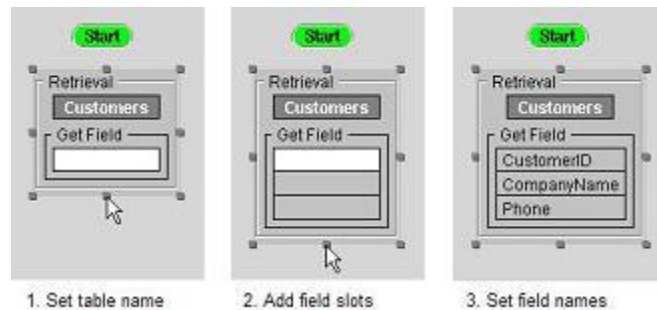


When a component lands in the Workspace after dragging, it gets selected (control points are shown around it). While the Retrieval component is still selected, set the name of the table by clicking on Customers entry in the Palette's Source column (alternatively, you can drag it into the Retrieval component's table name area at the top of the box).

Resize the Retrieval component by dragging its bottom control point down until there are three field slots, then click on these fields in the Palette's Fields column:

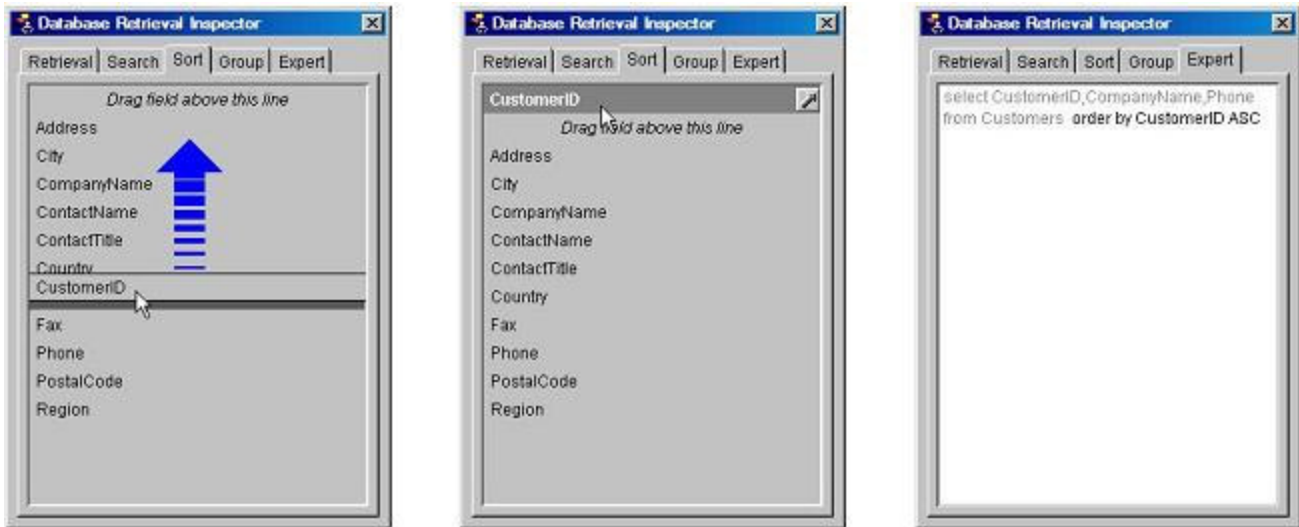
- CustomerID
- CompanyName
- Phone


(or, like the table name, you can drag the field names into their slots). This sequence is shown in the figure below:



The Retrieval component, as set here, instructs the machinery of Scribe to generate an SQL command that will, at the run-time, fetch CustomerID, CompanyName and Phone values from all rows that are found in the Customers table, and place them into the variables named Customers.CustomerID, Customers.CompanyName and Customers.Phone (the "Customers" qualifier of the variables' names is implied in the Retrieval box, the fully-qualified names will be used in other components that make use of the table data).

We also want to sort the retrieved data by CustomerID. Sorting, filtering, grouping and other operations that make part of the table retrieval process, are set in the Database Retrieval Inspector window that fine-tunes the Retrieval component. With the Retrieval component still selected, click on Sort tab and drag up the entry named CustomerID until it reaches the top of the Inspector's scrollable view:



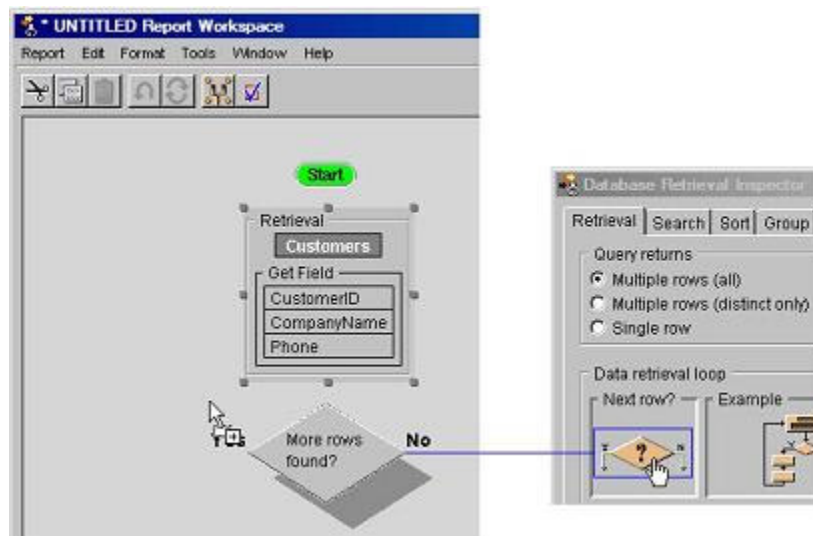
The arrow button  sets the order of sorting (ascending or descending).

The Expert tab shows the resulting SQL statement that Scribe will use to retrieve data from the Customers table. Note that the Expert view is partially editable (the portion of the text that is in black color). This allows the user to manually edit the SELECT statement if necessary. [Needless to say, such manual intervention should only be done by users skilled in SQL, which is implied by the tab caption: "Expert".]

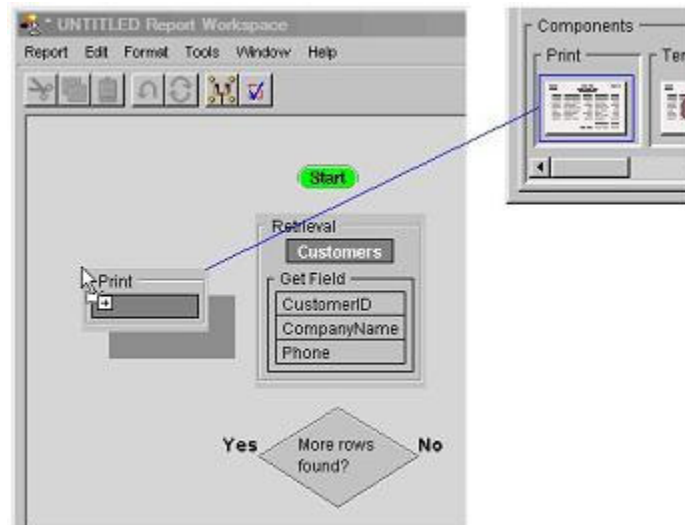
When the Retrieval component executes the first time, it fetches all table rows that match the Search criteria (in our case, all rows since the Search criteria is not set). The rows are placed into an internal stack inside the Retrieval component, and later dispensed to the procedure in every pass through the retrieval loop by the "Next row?" component (see below). The rows are dispensed in the FIFO order.

The next component to be added to the procedure is the "Next row?" component located in the Retrieval tab of the Database Retrieval Inspector. This component inspects the Retrieval stack, and if it contains data, the "Next row?" component extracts the field values from the topmost row, places them into the procedure variables, and removes the row from the stack. This is typically done in a loop, and the process continues until the Retrieval stack is empty.

Drag the "Next row?" component from the Inspector and place it under the Retrieval component as shown in the following figure:

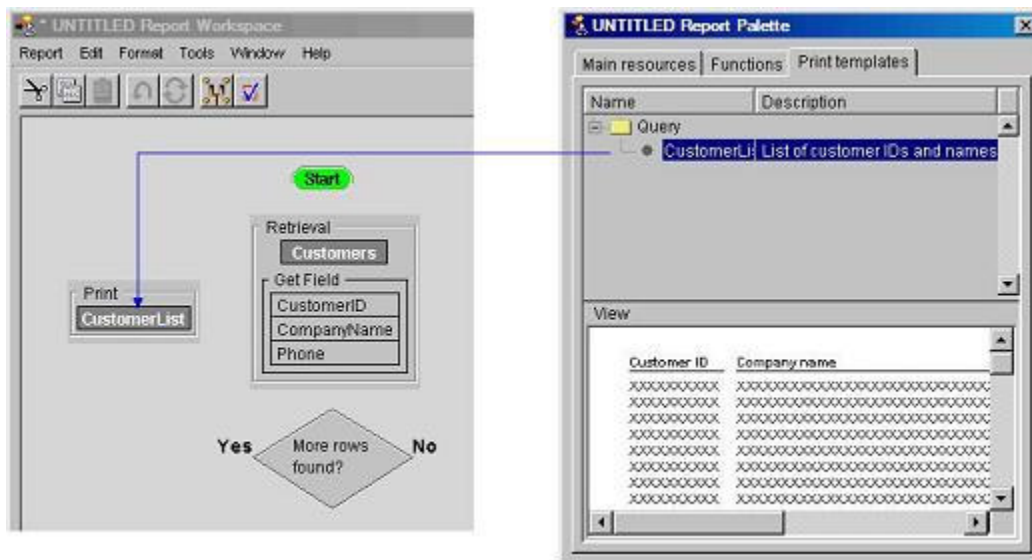


We are now ready to add the last - but crucial - component to the Report Procedure, namely the Print Template that was created in Tutorial1. In the Report Procedure, Print Templates are represented by Print components. Drag the Print component, located in the bottom left corner of the Palette, to the Workspace as shown in the figure below:

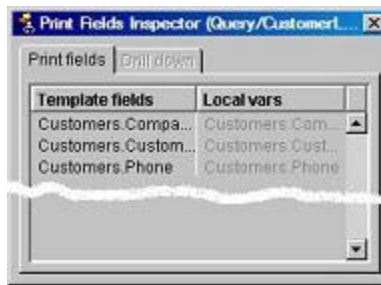


When the Print component lands, the Palette automatically selects "Print templates" tab and shows the list of available Print Templates. Click on the CustomerList entry; the View box at the bottom of the Palette will display the preview of the selected template. Click on it again, and the Print component is set to this template. (You can skip the preview step by dragging the print template name from the Palette straight into the Print component.)

This process is illustrated in the following figure:



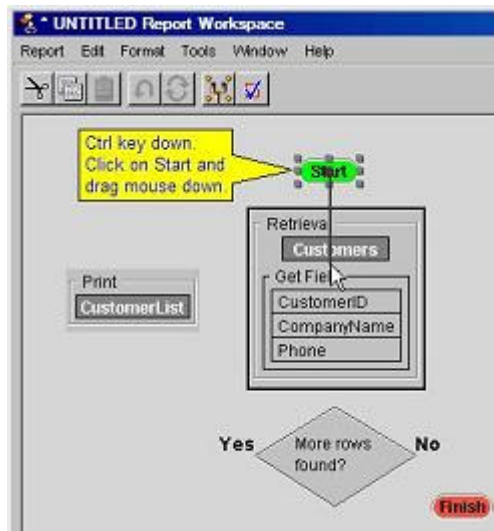
The Print Template carries a number of print fields that display a dynamic content, i.e. fields that expect to get their data from the report procedure at the run-time. The data would come from the procedure variables (such as table fields set by the Retrieval component); this flow of data from the procedure to the print template is established in the Print Fields Inspector, shown below:



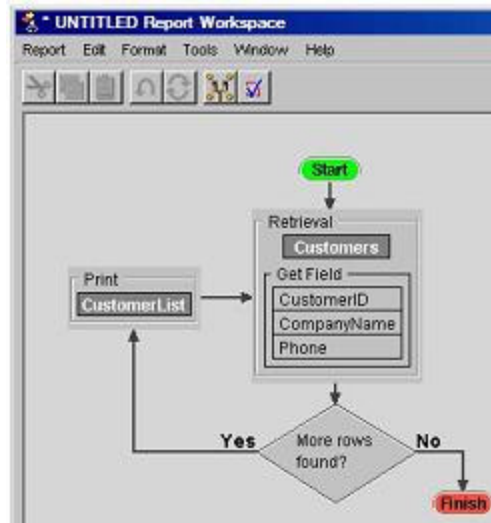
Initially, the "Local vars" column of the Inspector contains the same field names as in the "Template fields" column (the latter are the actual field names used in the print template). The program variables that carry data, should be dragged from the Palette and dropped onto the respective slots in "Local vars" column, thus linking the two worlds together. However, procedure variables that have the same names as their Print Template counterparts do not need to be dragged in: the links will be resolved automatically. This is the reason why we used the database table fields, rather than dynamic print fields, in assembling the CustomerList template: it relieves us from having to think about mapping the procedure variables to the template fields.


One last component to be brought into the picture is the Exit; it is already in the Workspace (it was put there by default, when the new Workspace window was opened), all we have to do is drag it closer to the "No" exit of the "Next row?" component.

Now is the time to set the order of the components' execution (control flow) by connecting the components. We do this by holding the Ctrl key down, and click-dragging the mouse from the source component to the target. Once the target component is reached, it will show a black outline; letting the mouse go at this point will fix the connecting line in the current position.




Setting the desired orientation of the connecting line takes some skills; the technique is described in Appendix A. Once the connections are set, the resulting procedure should look like this:



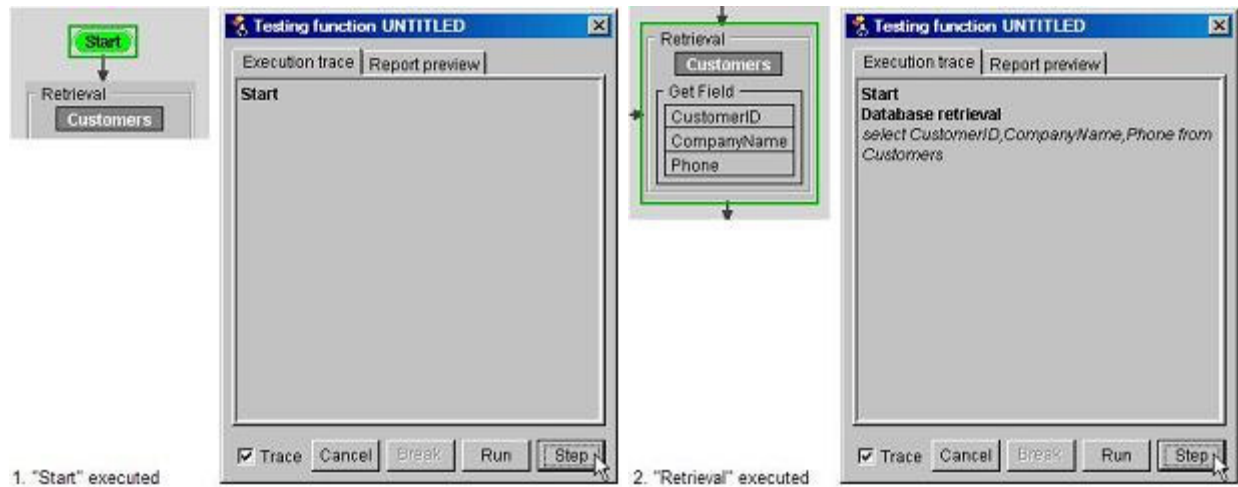
To verify that there are no errors in the procedure, press  button. Scribe runs a thorough check making sure that all executable components are connected with each other, that variables used in assignments are set to a valid content, etc. In case of errors, the user is given a visual diagnostic feedback. If you followed this Tutorial, there should be no errors, and the syntax check will return a "No errors" message that will be briefly displayed in the center of the Workspace.

However, the syntax check can expose only the simple, obvious errors and inconsistencies of the design. The real check is in running the report procedure and seeing that it produces the expected report ("the proof of the pudding..."). Scribe provides a graphical mechanism for running the procedure in test mode step-by-step, inspecting the results of each step and finding errors as they happen.

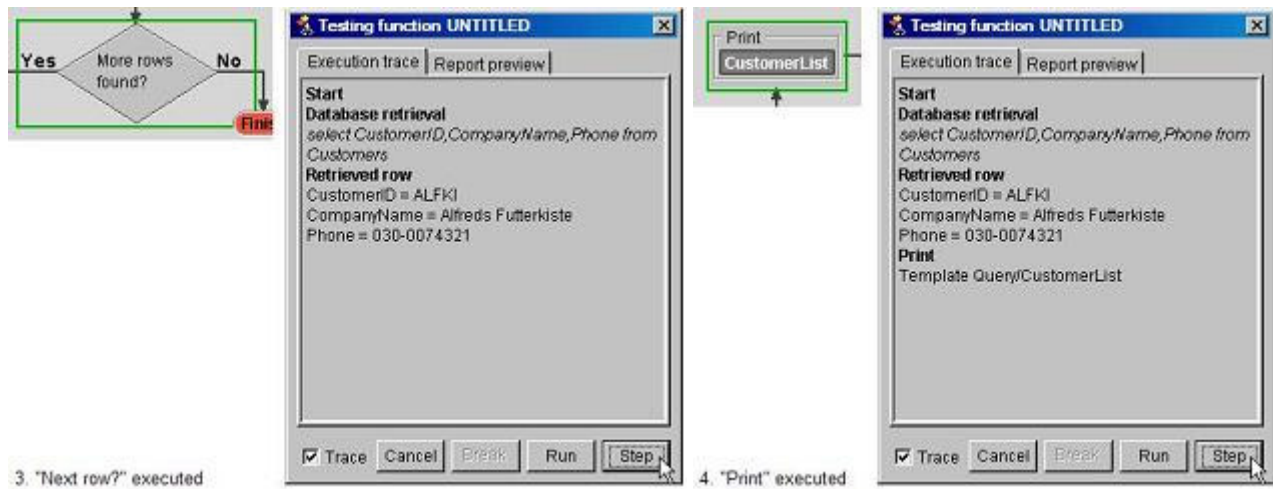
To begin the test, press this button: . Scribe will bring up the Test window and await your actions. At this point, you can:

- step through the procedure, executing one component at a time
- run the procedure without stopping before executing each component
- set a break on a selected component (i.e. select it first by clicking on it, then press the Break button), run the procedure until the breakpoint is reached, then step through it

In this Tutorial, we will start with stepping through the procedure. Press the Step button, and watch the execution trail:



At this point, the SQL SELECT statement has been executed, and the table rows have been returned and stored in the Retrieval component stack. The next step will bring back the first of these rows:

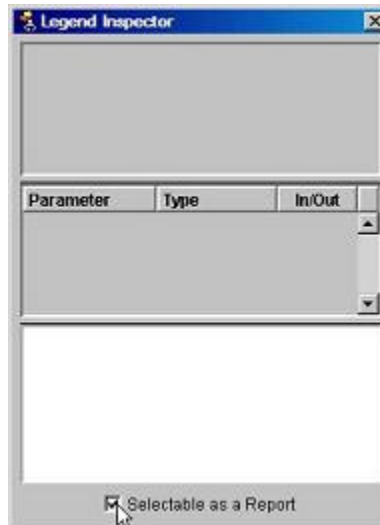


Our Print Template was executed for the first time. It is worth stopping at this point and taking a look at the "Report preview" tab. The Print Template should have printed its header (the first-time call) and the body (printed on every call, including the first one). This is what you should now see in the report preview:



Scribe Report Procedures can be used as "black box" components that can be plugged into other reports or

processes, rather than running as primary reports. This ability to compartmentalize reports helps keep complex reports manageable ("divide and conquer"), and makes it possible to reuse blocks of graphical flowcharts. To tell Scribe that the procedure under construction is intended to be run as a report (rather than being a building block for some other procedure of larger scope), we have to set the flag in the procedure's Legend Inspector (click anywhere on the Workspace's background, or on the component that does not have its own Inspector, like Start, Finish or Hub, to bring up the Legend Inspector):



It is now time to save the report procedure. Scribe uses a hierarchical naming convention for the report procedures, similar to the one used for print templates. We will name this procedure "Query/CustomerList". It is the same name used for the print template in Tutorial1, but it is acceptable since procedures and print templates "live" in different universes, and one cannot be confused with another.

Select Report > Save... menu item, and type in the report name into the panel as in the figure below:

