Printing with 4D View

By Larry Sharpe

Technical Note 07-11

Abstract

This Technical Note discusses using the printing commands from the 4D View plug-in for printing 4D View plug-in areas. A sample database is provided.

Overview

Several of my clients have asked for a way to easily print the data shown in the output forms that were created using 4D View. While I have shown them how to use the Reports function built into 4D and included it in their database, they still requested a simple way to print all the records shown, exactly as the data was shown in the window. "Click-and-Print" so to speak.

Since I have written a Technical Note or two in the past on using 4D View and I use that code for my own projects I thought it would make sense to update one of those Technical Notes and show how I am using the Printing commands from 4D View to accomplish this. I also included a few notes about upgrading the older Technical Note used here so that it works with the current release of 4D View for 4D 2004.5.

The Sample Database

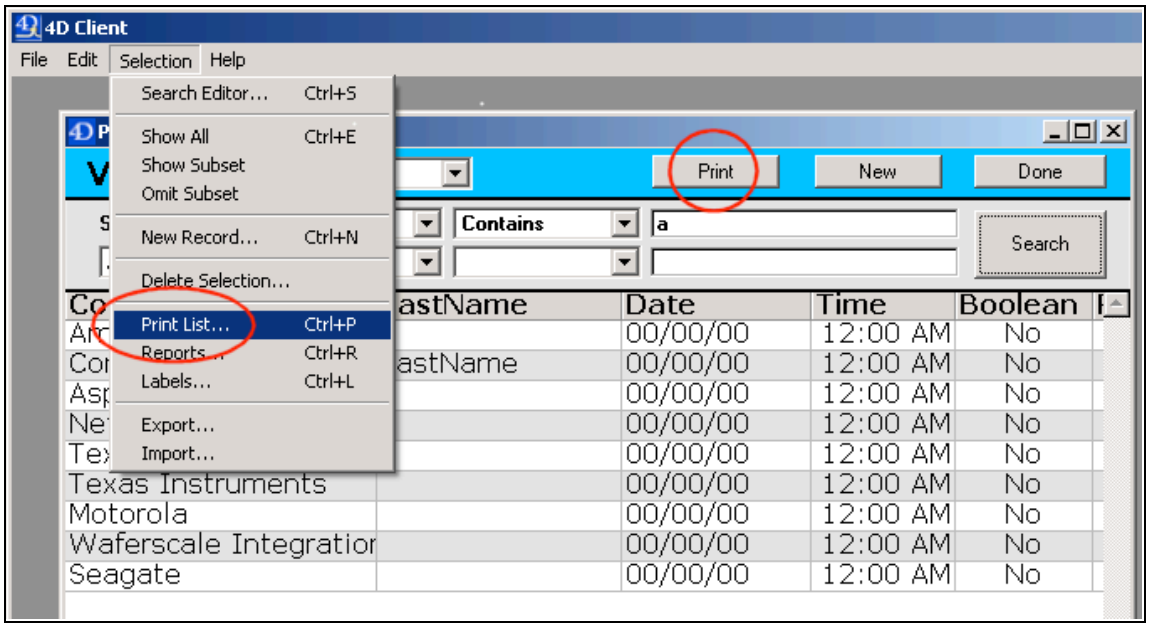
Most of this code was written for the Technical Note 02-45, "User Definable Views Using 4D View" published by 4D, Inc. on September 30, 2002. For more information on the example database and how the 4D View settings are configured, stored, and used please see the older Technical Note. In this example database all the places that I changed or added code to the original code have been marked with a `*updated 02/15/07* comment to make it easy to find those changes.

Also, 4D View's commands have been updated since Technical Note 02-45 was released. One important change is that column sorting is now done with a specific 4D View command, **PV SORT COLUMN**, rather than part of the **PV SET COLUMN HEADER** command. This update means that you can specify or change which column to sort on and how they are sorted without having to setup or change the column headers at the same time. The appropriate code changes have been made in the ***Views_4DV_Events*** and ***BuildViews_Fields_4DV_Format*** project methods.

Now on to the main function of this Technical Note, printing 4D View areas.

Printing 4D View Areas

As mentioned before, the example database included is pretty much the same as the previous version with the addition of a **Print List** menu item and a **Print** button found when looking at the "Views" window:



The menu item and the button each call the same Project method, named ***xOutput_Print4DView***. This method checks to see if you have the Alt (Windows) or Option (Mac OS) key down when clicked and calls the appropriate method as needed.

Holding no key down when clicking the button will print using the project method ***Views_4DV_Print_CodeDefined***, which means that the setup and management of the 4D View print settings are handled using 4D code.

Holding the Alt/Option key down while clicking the button will use the ***Views_4DV_Print_UserDefined*** project method, which allows the user to edit many of the programmable features in a dialog before printing.

Each of these two project methods is described below and each has comments in the code to help understand what they are doing.

Views_4DV_Print_UserDefined

Below is all the code for this method. Other than setting the Row Header to 1 pixel in width, we pretty much let the user set all the print options and page settings before printing. We also allow the user to print to an on screen preview page if they would like. I found while testing the various options that this was a great way to

save paper and ink, and was much faster than printing it to paper with my slow printer.

```
   `SET THE NAME OF YOUR 4D VIEW AREA TO BE PRINTED
C_LONGINT($area)
$area:=eViewsOutput4DV    `SAME 4D VIEW AREA BEING DISPLAYED

   `HIDE THE ROW HEADER
PV SET AREA PROPERTY  ($area;pv row headers width ;1)

   `ALLOW THE USER TO SET THE PRINT OPTIONS THEY DESIRE
PV EXECUTE COMMAND  ($area;pv cmd file printing options )
If (OK=1)
      `STANDARD PAGE SETUP DIALOG
   PV EXECUTE COMMAND  ($area;pv cmd file page setup )
   If (OK=1)
      If (Macintosh option down)
            `PRINT TO SCREEN, DON'T WASTE PAPER, INK WHILE TESTING
         PV EXECUTE COMMAND  ($area;pv cmd file print preview )
      Else
            `PRINT TO PAPER
         PV PRINT  ($area)
      End if
   End if
End if
```

A nice feature of using PV EXECUTE COMMAND with the "pv cmd file printing options" parameter is that it remembers the settings from the last time it was used in this process. Note that closing the process (window) and opening a new process resets all the settings to a blank state.

You could use this command in the next example method after you have set the values you want in code, this would allow the user to change settings on the fly from the default settings which you coded.

Views_4DV_Print_CodeDefined

This method is where you can set most of the options needed to print your 4D View area using code rather than having the user set those options. The code for this method is roughly three pages long so it is not included here. The code itself is fully commented in the sample database. Additionally I will describe three important items about here:

- The first is that you can change the font, size and color of the printed data using code, (not the headers or footers, those are fixed) but the rows and columns of your data. Since we are using the same 4D View area to print as well as display our data I made a slight addition to the previous version of the *x4DView_Fields* method. Now you can pass an optional third parameter to this method that will let you set the font, size and color of the 4D View area before printing. As configured, the settings for displaying a 4D View area are set by default and the

method does not need to be passed the third parameter. If needed you could also modify this update to set these items for displayed 4D View areas as well. Just add more settings in the CASE OF section of this method as needed and pass it the name of your settings when setting up your 4D View areas.

- The second is that this method will check to see how wide your columns are and automatically set the print orientation to either Portrait or Landscape as needed. It will also tell you if the columns are too wide to print on one page. 4D View will print to multiple pages depending on both column width and number of columns, but I found that it was better if the data only printed one page in width. When trying to align the multiple-page-wide printout, the pages did not always get combined (by the users, not the printer) in the proper order and the resulting "view" of the data was then incorrect. Depending on your needs (and printer) you may want to modify the lines of code where the $pageWidth variable is checked in order to set the page orientation and limits of your printer and your users request.
- The third feature of this code is that you could print to a PDF document instead of a printer or preview page. In this example database I have commented out the few lines of code that you could use if so desired. Note that the commands are slightly different when using Macintosh or Windows computers and each of those settings can be set to either a specific document path or allow the user to enter the file name and path to use.

Do not forget that you could add the line PV EXECUTE COMMAND ($area;pv cmd file printing options) shown in the previous example method. You would add it right after you have set the values you want, and right before the PV EXECUTE COMMAND ($area;pv cmd file page setup) command. This would allow the user to manually change the settings on the fly, if needed, from the default settings that you coded.

Conclusion

With this Technical Note I have used several features of the 4D View plug-in:

1. Described how to use the 4D View Printing commands to print your data. This was in the *Views_4DV_Print_CodeDefined* and *Views_4DV_Print_UserDefined* methods.

2. Updating the older Technical Note to support the new PV SORT COLUMN command in 4D View. This updated the *Views_4DV_Events* and *BuildViews_Fields_4DV_Format* methods.

3. Showed how to modify the existing *x4DView_Fields* method to easily change the font, size and color of the printed or displayed 4D View area by passing an optional third parameter.