

TablePrinter

Inherits From: Object
Declared In: TablePrinter.h

Class Description

TablePrinter is a high level interface to DBTableViewPrinter, a class which nicely formats and prints DBTableViews. Each instance of TablePrinter manages its own off screen instance of DBTableViewPrinter. TablePrinter contains methods for formatting and printing a DBTableView. Additionally, direct access to the DBTableViewPrinter instance can be attained by querying the TablePrinter for the **printerView**.

Although DBTableViewPrinter could be instantiated directly, it is much cleaner to let TablePrinter handle it. Since DBTableViewPrinter is a view which can print itself, it must exist in a window and be resizeable. TablePrinter manages it's own off screen window so

that you don't have to. It also contains methods and outlets to allow easy usage in Interface Builder.

TablePrinter was designed to be used as a palette. For simple operation, just drag a TablePrinter into your nib and connect its *tableView* outlet to the desired DBTableView. Then make the new TablePrinter instance the target of the print button and **printPSCode:** the action.

Setting Parameters by Responding to User Action

The TablePrinter inspector can be used to alter the print parameters. Additionally, user interface objects like switch Buttons and ColorWells can target the TablePrinter object to change print parameters at run time. For example, you could build an accessory view to the PrintPanel which lets the user adjust the look of the output.

There are three types of user action methods. First, methods that respond to switch Buttons (or any object that responds to **±state**) have names ending with the word "Toggle". Second, methods that respond to ColorWells (or any object that responds to **±color**) have names ending with the word "Color". Finally, all remaining methods set either a character, string, or number, and respond to TextFields (or objects which respond to **±intValue** or **±stringValue**).

For example, to allow the user to toggle page number printing On/Off, drag a switch Button into the nib and drag a connection from it to the TablePrinter instance. Select the method **-pageNumbersToggle:**. To allow the user to change the color of the page number, drag a ColorWell into the nib and drag a connection from it to the TablePrinter instance. Select the method **±pageNumColor:**. To allow the user to specify the starting page number, drag an editable TextField into the nib and drag a connection from it to the TablePrinter instance. Select the method **±startPageNum:**. It would also be nice if you initialized the values in these user interface objects to reflect the values set in the TablePrinter by the inspector.

Required Usage of TablePrintPanel

The TablePrintPanel object included with TablePrinter Palette is a subclass of PrintPanel which allows the setting of pages per row, so that it can adjust the total pages sent to the printer to reflect compound pages. For instance, if you ask to print from 2 to 2 on a table that is three pages wide, It will set PrintInfo to print 4 through 6. **It is essential that TablePrintPanel be the first PrintPanel referenced anywhere in an application which uses a DBTableViewPrinter.** This is because an Application only has one PrintPanel object. If the regular PrintPanel object gets referenced (and subsequently loaded) first by another place in an application, then attempts by DBTableViewPrinter to message TablePrintPanel will fail. If TablePrintPanel is loaded first, all other references to PrintPanel will work without a hitch, as if there were no change. A simple message like:

```
[TablePrintPanel new];
```

in the **± appDidInit:** method will solve this problem.

Instance Variables

```
id tableView;  
id printerView;
```

tableView	An outlet to the DBTableView.
-----------	-------------------------------

printerView	An outlet to the DBTableViewPrinter.
-------------	--------------------------------------

Method Types

Initializing and freeing

- ± init
- ± free

View access

- ± setTableView:
- ± tableView
- ± printerView

Printing

- ± printPSCode:

Responding to user action

- forceColorToggle:
- ± gridLinesToggle:
- ± headerBezelToggle:
- ± pageNumbersToggle:
- ± pageNumSeparator:
- ± rowNumbersToggle:
- ± rowNumHeader:
- ± selectedRowsToggle:
- ± startPageNum:
- ± startRowNum:
- ± headerTextColor:
- ± headerBackColor:
- ± rowNumTextColor:
- ± rowNumBackColor:
- ± gridColor:
- ± pageNumColor:
- ± backgroundColor:

Archiving

- read:

- write:

Instance Methods

backgroundColor:

± backgroundColor:*sender*

Queries *sender* for **color** and uses the return value to set the color of the main background. The default is NX_COLORWHITE. Returns **self**.

See also: - **setBackgroundColor:** (DBTableViewPrinter)

forceColorToggle:

± forceColorToggle:*sender*

Queries *sender* for **state**. If state is YES, then the TablePrinter will render color output, regardless of whether the chosen printer device can display color. The default is NO.

See also: - **setForceDrawColor:** (DBTableViewPrinter)

free

- **free**

Frees the instance variables, the DBTableViewPrinter it manages, and itself.

gridColor:

± gridColor:*sender*

Queries *sender* for **color** and uses the return value to set the color of the grid. The default is NX_COLORBLACK. Returns **self**.

See also: ± **gridLinesToggle:**, ± **setGridLinesColor:** (DBTableViewPrinter)

gridLinesToggle:

± gridLinesToggle:*sender*

Queries *sender* for **state**. If state is YES, the TablePrinter will print grid lines between rows and columns. The default is NO. Returns **self**.

See also: - **setGridLinesOn:** (DBTableViewPrinter)

headerBackColor:

± headerBackColor:*sender*

Queries *sender* for **color** and uses the return value to set the color of the background in the column headers. The default is NX_COLORDKGRAY. Returns **self**.

See also: - **setColumnHeadersBackColor:** (DBTableViewPrinter)

headerBezelToggle:

± headerBezelToggle:*sender*

Queries *sender* for **state**. If state is YES, the TablePrinter will print column headers with

bezeled borders. The default is NO. Returns **self**.

See also: - **setColumnHeadersBezeled:** (DBTableViewPrinter)

headerTextColor:

± headerTextColor:*sender*

Queries *sender* for **color** and uses the return value to set the color of the text in the column headers. The default is NX_COLORWHITE. Returns **self**.

See also: - **setColumnHeadersTextColor:** (DBTableViewPrinter)

init

- **init**

The designated initializer for this class. Returns **self**.

pageNumColor:

± pageNumColor:*sender*

Queries *sender* for **color** and uses the return value to set the color of the page number text. The default is NX_COLORBLACK. The page number background is always clear (hence, the color of the background). Returns **self**.

See also: **± pageNumbersToggle:**, **± backgroundColor:**, **± pageNumberColor** (DBTableViewPrinter)

pageNumbersToggle:

± pageNumbersToggle:*sender*

Queries *sender* for **state**. If state is YES, the TablePrinter will print a page number on each printed page. The default is NO. Returns **self**.

See also: - **setPageNumbersEnabled:** (DBTableViewPrinter)

pageNumSeparator:

± pageNumSeparator:*sender*

Queries *sender* for **stringValue** and uses the first character of the return string as the character which separates page numbers of compound pages. The default is '-'. If a DBTableView is too wide to fit on a single page, it will be spread across multiple pages. Therefore, page 1 would become page 1-1, 1-2, 1-3, etc. Returns **self**.

See also: - **pageNumbersToggle:**, - **setPageNumberSeparator:** (DBTableViewPrinter)

printerView

± printerView

Returns the DBTableViewPrinter object being managed by this TablePrinter. The DBTableViewPrinter can be messaged directly to set print options.

read:

- **read:**(NXTypedStream *)stream

Unarchives a TablePrinter instance from *stream*.

rowNumBackColor:

± **rowNumBackColor:***sender*

Queries *sender* for **color** and uses the return value to set the color of the background in row numbers. The default is NX_COLORWHITE. Returns **self**.

See also: ± **rowNumbersToggle:**, ± **setRowNumbersBackColor:** (DBTableViewPrinter)

rowNumberHeader:

± **rowNumberHeader:***sender*

Queries *sender* for **stringValue** and uses the return string as the title for the row number column. The default is "#", but any string up to 50 characters can be used. Returns **self**.

See also: ± **rowNumbersToggle:**, ± **setRowNumbersTitle:** (DBTableViewPrinter)

rowNumTextColor:

± **rowNumTextColor:***sender*

Queries *sender* for **color** and uses the return value to set the color of the text in row numbers. The default is NX_COLORBLACK. Returns **self**.

See also: \pm **rowNumbersToggle:**, \pm **setRowNumbersTextColor:** (DBTableViewPrinter)

rowNumbersToggle:

\pm **rowNumbersToggle:***sender*

Queries *sender* for **state**. If state is YES, the TablePrinter will print a column of row numbers on each printed page. The default is NO. Returns **self**.

See also: \pm **setRowNumbersEnabled:** (DBTableViewPrinter)

selectedRowsToggle:

\pm **selectedRowsToggle:***sender*

Queries *sender* for **state**. If state is YES, the TablePrinter will only print the currently selected rows from the DBTableView. The default is NO. Returns **self**.

See also: \pm **setSelectedRowsOnly:** (DBTableViewPrinter)

setTableView:

\pm **setTableView:***tableView*

Sets the DBTableView to be printed to *tableView*. When using a TablePrinter object, always use TablePrinter's **setTableView:** method rather than DBTableViewPrinter's **setTableView:** method. Returns **self**.

See also: \pm **setTableView:** (DBTableViewPrinter)

startPageNum:

± startPageNumber:*sender*

Queries *sender* for **intValue** and uses the return value as the starting page number. The default is 1. Note: this value is used for display purposes only and has no bearing on which pages are actually printed. It is still necessary to enable page number printing. Returns **self**.

See also: **± pageNumbersToggle:**, **± setStartingPageNumber:** (DBTableViewPrinter)

startRowNum:

± startRowNumber:*sender*

Queries *sender* for **intValue** and uses the return value as the starting row number. The default is 1. Note: this value is used for display purposes only and has no bearing on which rows are actually printed. It is still necessary to enable row number printing. Returns **self**.

See also: **± rowNumbersToggle:**, **± setStartingRowNumber:** (DBTableViewPrinter)

tableView

± tableView

Returns the DBTableView to be printed or **nil** if not set..

See also: **± tableView:** (DBTableViewPrinter)

write:

- **write:**(NXTypedStream *)stream

Archives a TablePrinter instance to *stream*.