The Browse System

A new Browse system has been implemented in Clipper 5.0 which uses the object data type. The Browse system is made up of two classes: TBrowse and TBCOLUMN which are discussed in this section. For a complete list of the exported instance variables and methods, refer to the Standard Classes chapter in this book.

The TBrowse Class

What is a TBrowse Object

A TBrowse object is a general purpose browsing mechanism for table-oriented data. TBrowse objects provide a sophisticated architecture for acquiring, formatting and displaying data. Data retrieval and file positioning are performed via user-supplied code blocks, allowing nearly unlimited flexibility and control. Display formatting is completely controllable through the use of user-specified formatting strings.
A T Browse object evaluates supplied code blocks to retrieve data. The data is organized into rows and columns, and displayed within the confines of a specified rectangular area on the screen. Programmable behaviors allow navigation of the data. New data is automatically retrieved as required by navigation requests. If the cursor is moved past the edge of the visible rectangle, any rows or columns that exist beyond that edge are automatically brought into view.

Creating a T Browse Object

The syntax for creating a T Browse object is as follows:

```clip
browse := TBrowseNew(<Top>, <Left>, <Bottom>, <Right>)
```

This line of code creates a new T Browse object and stores it into the variable `browse`. The coordinates define the rectangular area for the T Browse object.

An additional T Browse class function is T BrowseDB(). This function has the same syntax as T BrowseNew() but it is designed for use when browsing a database file. T BrowseDB() creates a T Browse object with default code blocks for data source positioning within database files. The default code blocks execute the GO TOP, GO BOTTOM, and SKIP operations.

Defining Columns

The columns of a T Browse object consist of T BColumn objects. Each T BColumn object contains the information necessary to define a single column of a browse (see The T BColumn Class section below).

In the example below, a new T BColumn object is created and added to an existing browse object.

```clip
column := T BColumnNew("Name", [11 name])
browse:addColumn(column)
```

The T BColumnNew() function creates a new column object. The T Browse:addColumn() message is then sent to the T Browse object to add a new column into the browse. Columns can be added, modified, or rearranged at any time.

Color Definitions

A T Browse object uses a color table to control the display attributes of displayed data. A color table is an ordered list of Clipper color specifiers. By default, a T Browse object will use as its color table the five colors defined by the standard SETCOLOR() function. If desired, a more complex color table can be specified by assigning a valued to T Browse:colorSpec.

There are three ways to change color in a T Browse object:

- Supply a new color table using T Browse:colorSpec. This affects the color of all cells.
Assign TColumn:defColor for one or more TColumn objects. This affects the color of an entire column, including headings and footings (see the section on the TColumn class).

Assign a color control code block to TColumn:colorBlock for one or more of the TColumn objects. This allows the color of individual data values to be controlled (see the section on the TColumn class).

All colors are specified using indexes into the color table in use by the TBrowse object. If no custom color table is defined using TBrowse:colorSpec, the color indexes are limited to 1 through 5, corresponding to the five colors in the standard SETCOLOR() function.

Stabilization

The operations necessary to properly display the TBrowse data include positioning the data source, retrieving the data (via the code blocks supplied in the TColumn objects), formatting the data, and physically updating the display. These operations are collectively referred to as stabilizing the TBrowse object. When a TBrowse is stable, it means that all rows and columns are correctly displayed, that the database or other data source is positioned to the current row, and that the current cell is highlighted.

In order to avoid response delays due to potentially lengthy data retrieval operations, stabilization occurs in small increments. The stabilize method performs a small part of the stabilization process each time it is invoked. It returns true (T.) if stabilization is complete; otherwise it returns false (F.). This allows the stabilization process to be interrupted in response to a keypress. The TBrowse:stabilize() message is usually sent in a loop like this:

```
DO WHILE (.NOT. browse:stabilize()) ; ENDDO
```

The example above forces the complete stabilization of the TBrowse object. As mentioned above, however, it is often desirable to allow the user to interrupt the stabilization. This is done by simply checking for a keystroke during the stabilization loop:
This technique allows the user to move to the next screen full of data without waiting for the current screen to fill up.

The TBColumn Class

What is a TBColumn Object
A TBColumn object is a simple object containing the information needed to fully define one column in a TBrowse object (see the section on the TBrowse class). TBColumn objects have no methods, only exported instance variables.

Class Function
The function associated with the TBColumn class is as follows:

TBColumnNew(<Heading>, <Block>)

TBColumnNew() returns a new TBColumn object with the specified values for TBColumn:heading and TBColumn:block. Other elements of the TBColumn object can be assigned directly using the syntax for assigning exported instance variables.