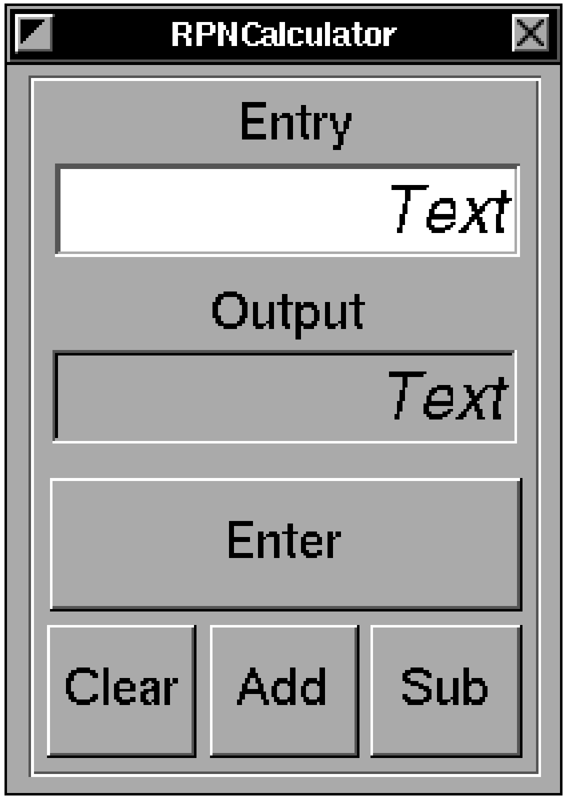
OOP Class
IB_RPNCalculator

Purpose: In this lab you will create a custom object and connect it to an interface in IB. The lab will use the RPNCalculator object previously desgned and modify it to work with a simple screen interface.



1. Start PB and begin a new application (**Project**/**New...**), name it **RPNCalculator**.

2. In the PB file window, choose **Interfaces**/**RPNCalculator.nib**. Open it in IB and design the calculator interface as shown above.

3. Save the interface (**Document**/**Save** or **command-s**).

4. Add the *RPNCalculator.(hm)* files from the solution to the **RPNCalculator** lab by selecting **Classes** in the PB file window, double-clicking its suitcase, and finding those files in the Open panel. They will be copied to the new project directory. You will also need *Stack.h* and *Stack.o*. To get *Stack.h*, select **Headers** in the PB file window and double-click its suitcase; to get *Stack.o*, select **Other Sources** and double-click its suitcase. Save the project (**Project**/**Save**).

5. After the files are copied, modify the *RPNCalculator.(hm)* files as follows:

- add **entryField** and **display** instance variable outlets
- create a new method **clearEntry** which simply writes a blank to the entry field and then reselects it.
- modify method **(float)enter:(float)number** to be **enter:sender**. This method should get the value to be entered from **entryField**, enter it, write the top of the stack to the display, and then clear the entry field. It is called either by pressing the Enter button or pressing Return in the Enter field.
- modify method **(float)add:(float)number** to be **add:sender**. This method should simply call the **add** method to add the top two values on the stack, then write the result to the display and clear the entry field. It is called by pressing the Add button.
- modify method **(float)subtract:(float)number** to be **subtract:sender**. Similar to **add:sender**. It is called by pressing the Sub button.
- modify the **allClear** method to be **allClear:sender**. This method should clear the stack, then write the top of the stack to the display and clear the entry field. It is called by pressing the Clear button.
- add an **appDidInit:** method which clears the entry field and the output display (this will be at program startup). The RPNCalculator object must be set as the nib File's Owner's delegate in IB.

Note: This calculator may not function quite the way you are used to in an RPN calculator. In particular, a value *must* be entered before it is considered for any arithmetic operation because the operations only look at values already on the stack. In other words, a value typed into the Enter field but not yet Entered (by pressing the Enter button or pressing return) is not seen for any subsequent add or subtract operation. This restriction is easily removed if you put digit buttons on the calculator as suggested in **Extensions** below.

6. In the IB Classes window, **Parse** (Operations pull down menu) the new RPNCalculator class.

7. Again from the Operations pulldown menu, **instantiate** an RPNCalculator. An icon for a generic RPNCalculator instance should appear in the File window.

8. By control-dragging, **connect** the interface objects appropriately to the RPNCalculator instance. In the case of the **entryField**, connect it to the **performClick:** method of the Enter button, then connect the button to the RPNCalculator instance. Finally, connect the **delegate** outlet of the **File's Owner** icon in the File Window to the RPNCalculator Instance. (The nib File's Owner is the Application object, which sends the **appDidInit:** message to its delegate, if it has one.)

9. Add the Stack class to the project. Because you only have the Stack.o file (and Stack.h) for the Stack class, not the Stack.m source, you should do the following in PB:

> Add Stack.h: double-click **Headers** in the browser and select Stack.h in the Open panel.
> Add Stack.o: double-click **Other Sources** in the browser and select Stack.o in the Open panel.

PB will copy these files into the project.

10. Now save the project, and build and run the program.

Extensions: Look at **NextDeveloper**/**Examples**/**AppKit**/**CalculatorLab**. See if you can enhance the user interface to this lab so that the user can click on digit buttons on the screen rather than enter the number through the entry textfield. In general, see if you can make RPNCalculator look more like CalculatorLab.