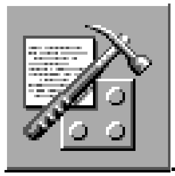
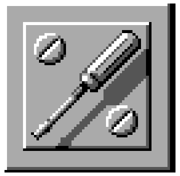
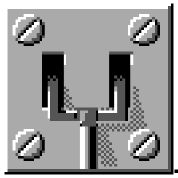
OOP Class Lab
Converter

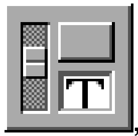
Purpose: In this lab you will design a simple calculator to convert kilograms to pounds. It is a simple application showing how to design a custom object in Interface Builder.

1. Start up **Project Builder** . Begin a new application by selecting **Project**/**New...** on the main menu. A panel titled New Project will pop up asking for the project name. Leave the popuplist labelled "Project Type:" in the Application position. Select the directory where you want the project to be, then type **Converter** and click OK. A project directory is created and the project file window pops up.

2. Design an interface as described below. Choose **Interfaces**/**Converter.nib** from the PB browser and double-click the icon. Interface Builder  is launched, and it creates a main nib file with a window named **My Window**.

Note: You can test the interface after each step below. Test by choosing **Document**/**Test Interface** on the IB main menu. Choose **Quit** in test mode to return to IB, or double-click on the IB icon which will change in test mode to look like: .

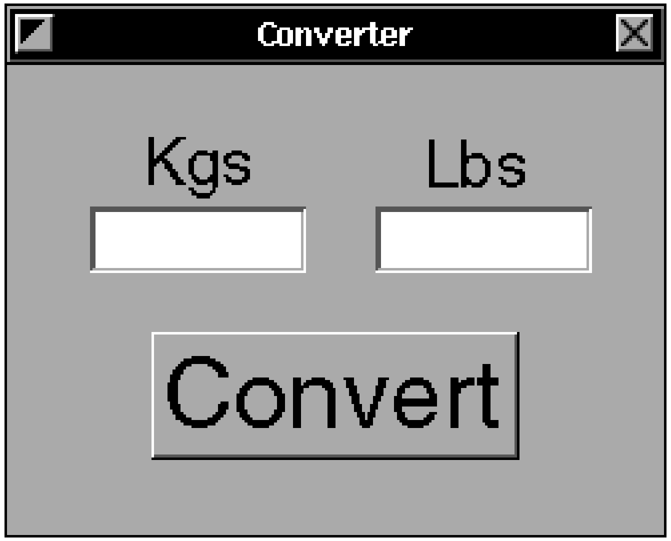
a. Click on your main window; its title bar should be black. Call up the Window Inspector by selecting **Tools**/**Inspector...** on the IB main menu. In the Window Inspector (Attributes), set the window so that it does not have a resize bar or a close control. Also change its title to **Converter** and press return.

b. From the third palette , drag two Text fields 

, two Title labels 

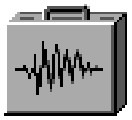
(actually, they are both instances of TextField), and a button 

into the window. Arrange them and set their attributes as shown below. Also click the temporary resize button in the upper left corner of the window, and then resize the window to the size shown.

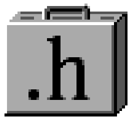


3. Make Connections.

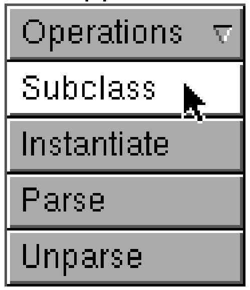
a. *Control-drag* and connect the TextField under the Kgs label to the Convert button. In the inspector, connect to the **performClick:** action method of the button.

b. Click on the Sounds suitcase  in the File Window to open it. Drag a desired sound out and drop it on the button.

4. Create the custom converter object.

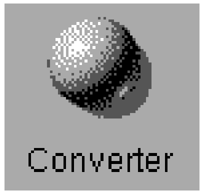
a. *Subclass Object*. Click the Classes suitcase 

in the File Window. In the Classes browser that appears, move to the left and select the **Object** class. In the Operations pull down menu



select **Subclass**. The Inspector should now become a Class Inspector, inspecting the new MyObject class. Double-click the MyObject text in the Class Inspector, rename it to **Converter**, then press Return. You are designing a custom class named Converter which is a subclass of Object.

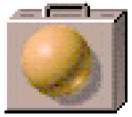
b. *Instantiate the Custom Object*. Go back to the Operations pull down menu and select **Instantiate**.

An icon representing an instance of Converter 

should appear in the File Window. The CustomObject Inspector on the right should show that it is a member of the Converter class.

c. *Add Outlets and Actions*. Now click on the Classes button again and select the Converter class in the browser. This should bring up the Class Inspector on the right. Select Attributes in the popuplist at the top. Choose the *Outlets* radio button. In the textfield provided at the bottom, enter an outlet named **input** and press return. Do also for another outlet named **output**. Now click the *Actions* radio button. In the same manner as for

Outlet, enter a new action method named **convert:**.

d. *Make Connections to Custom Object*. Click on the Objects suitcase  to display the Converter instance icon. *Control-drag* from the Converter Instance icon in the File Window to the textfield under the Kgs label in the main window. Make the connection for the **input** outlet. Do likewise for the **output** outlet to the field under the Lbs label. Now *control-drag* from the button in the main window to the Converter Instance. Make the connection to the **convert:** action method.

5. Save the IB Files. Select **Document**/**Save** in the main IB menu.

6. Unparse the Custom Object. Click on the Classes button in the File window, then select Converter in the class browser. Go to the Operations pull down menu and choose Unparse. When prompted, answer *Yes* to both *Create and Insert files*. IB will create the skeleton class files Converter.m and Converter.h, and insert them under the Classes and Headers sections of the browser in Project Builder.

7. Add Instance Variables to the .h file. In the PB window, choose Headers/Converter.h and open it (double-click it). There are no new instance variables to be added to the .h file, so you can close it again.

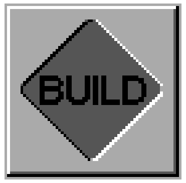
8. Add Code and #import's to the .m file. In the PB file window, choose Classes/Converter.m and open it. Add the code to the **convert:** method as follows:

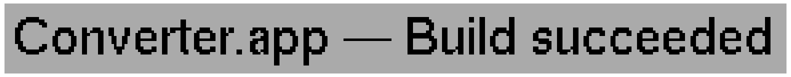
```
float weight;                      //declare local float variable
weight=[input floatValue];         //message input field to get value
weight=weight*2.2;                 //convert
[output setFloatValue:weight];     //message output field to set value
```

Place this just above "return self;" inside the braces. (Do you know how to combine the these into one line?) There are no new #import's to be added.

9. **Save the .m and .h files.** Do it!

10. Select Builder 

in the PB window and then select 

button to build the project. The PB gives a message like 

when it finishes properly. The **Converter.app** now is in the project directory. Double-click it to run it, or just click the Run button 

in PB. Test the program. Conversion should take place either when you hit return in the input field or when you click the button. You should hear your sound at each conversion.

Extension: Try modifying this program such that it can also do Celsius-to-Fahrenheit conversion. It will be easiest if you directly modify your solution to the version above as follows:

1. In Edit, open Converter.h and add two outlets, **inputLabel** and **outputLabel**, one per line, and one BOOL flag, **convertingTemp**, as new instance variables.

2. Add two new action methods, **selectWt:** and **selectTemp:**, to both Converter.h and Converter.m.

3. Write the code in Converter.m for the two new methods: when one of the two new methods is called, it should:

- change **inputLabel** and **outputLabel** (use **setStringValue:**) above the textfields to reflect the correct conversion.
- set **convertingTemp** appropriately so that the **convert:** method will know which kind of conversion to do.

4. Now also modify the **convert:** method to do either conversion, depending on the state of **convertingTemp**.

5. Go back to IB and add a submenu named **Conversion** to the main menu. Put two items in the submenu: **Kgs-to-Lbs** and **C-to-F** (copy and paste in the submenu to get the second item).

6. Now IB must be told of the changes to the Converter class interface. To do this, select **Parse** in the Operations pull down menu in the File window. In the Open panel that appears, select Converter.h to parse. In the Class Inspector you should see all of the new outlets and methods for the Converter class. (You will not see the **convertingTemp** instance variable since it is not an outlet to another object.

7. Control-drag from the Converter instance in the File Window to labels above the input and output textfields to connect the **inputLabel** and **outputLabel** outlets. Also control-drag from the two new menu items to the Converter instance to connect to the **selectWt:** and **selectTemp:** methods.

8.   Now save the nib file (**Document**/**Save**), return to PB and save the project (**Project**/**Save**), and build and run the program.