

High Level Containers – Option Dialogs



The most complex of all Dialogs, the Option Dialog is a combination of the other three Dialogs and has a few more settings which can be specified.

The full settings for the Option Dialog is as follows ...

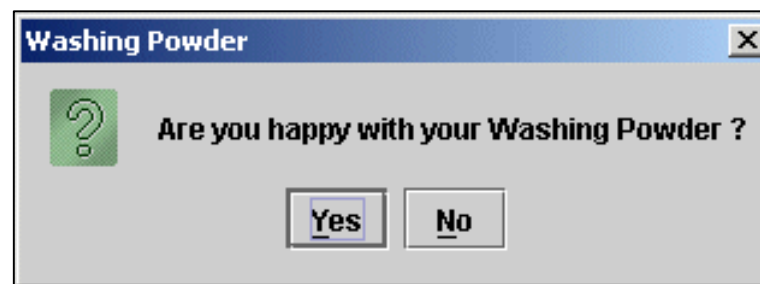
```
showOptionDialog(parent component  
                  ,The prompt in the box  
                  ,The title of the box  
                  ,Button options  
                  ,Message type  
                  ,Icon  
                  ,Button Array  
                  ,Default Button);
```

High Level Containers – Option Dialogs



The simplest usage of the Option Dialog is to simulate a Confirm Dialog, this involves setting the arguments not needed to null ..

```
int Names = JOptionPane.showOptionDialog(frame,  
    "Are you happy with your Washing Powder ?",  
    "Washing Powder",  
    JOptionPane.YES_NO_OPTION,  
    JOptionPane.QUESTION_MESSAGE,null,null,null);
```

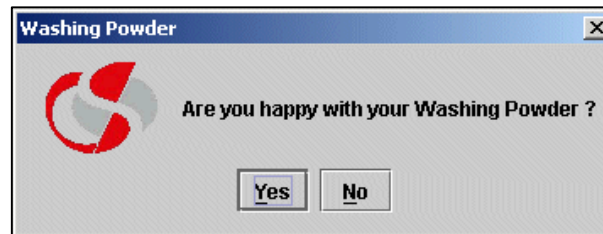


High Level Containers – Option Dialogs



This next version changes the icon of the Dialog to display the Seer logo, note the Developer is checking at the same time for the image files existence ...

```
ImageIcon Seericon = new ImageIcon("seer.gif");  
if (Seericon.getIconHeight() <= 0)  
{System.out.println("Expected image file not found");  
System.exit(0);}  
int Names = JOptionPane.showOptionDialog(frame,  
    "Are you happy with your Washing Powder ?",  
    "Washing Powder",  
    JOptionPane.YES_NO_OPTION,  
    JOptionPane.QUESTION_MESSAGE,  
    Seericon,null,null);
```



High Level Containers – Option Dialogs



Having customised the icon, message and dialog title, this next few slides deals with changing the option buttons that the user is presented with ... the Developer needs to create an array of String, each title of the Button will be derived from the value in the String.

In the following example the Developer has specified four positions in the String and these values will form the labels of the buttons, there is no limit to the number of values specified in the String, but bear in mind the width of the screen when being displayed, five seems to be a reasonable limit ..

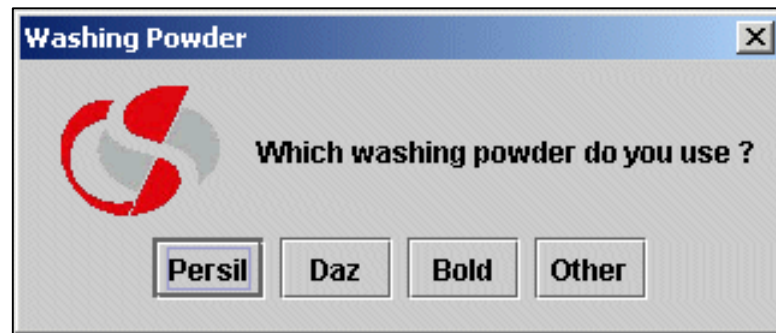
When creating buttons for a dialog, remember that their array position not only dictates their display position but also the integer they will return if that button has been selected ...

High Level Containers – Option Dialogs



```
import javax.swing.*;
public class swing7
{
    public static void main(String[] arguments)
    {
        JFrame frame = new JFrame("Title");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ImageIcon Seericon = new ImageIcon("seer.gif");
        String[] washing = {"Persil", "Daz", "Bold", "Other"};
        int Names = JOptionPane.showOptionDialog(frame,
                                                    "Which washing powder do you use ?",
                                                    "Washing Powder", 0, 0, Seericon,
                                                    washing, "Persil");

        System.exit(0);
    }
}
```



High Level Containers – Option Dialogs



Note in the previous example that the Developer no longer needed to specify the Message Type and Button Options because they have been customised in the other arguments passed, therefore 0 was passed in both cases.

In addition, the final argument was specifying the button which will have 'focus' when the Dialog is displayed, the Developer chose to specify the value of the button to be highlighted but could have specified washing[0] to specify its array position.