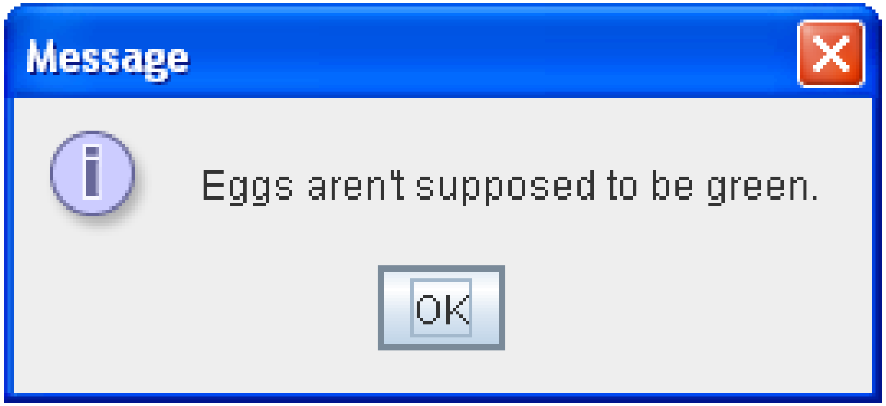
GUI Programming with Java

Session 9
Dialogs

- We will look at…
  - Creating Dialogues

- A dialog is basically a window that is more limited than frame.
- Several classes support *dialogs*.
- The ProgressMonitor class can put up a dialog that shows the progress of an operation.
- To bring up a print dialog, you can use the Printing API.
- To create custom dialogs, use the JDialog class directly.

- The code for simple dialogs can be minimal. For example, here's an informational dialog:



Eggs arent supposed to be green.

- Here is the code that creates and shows it:

```
JOptionPane.showMessageDialog(frame, "Eggs aren't supposed to be green.");
```

An Overview of Dialogs

- Every dialog is dependent on a frame.
- When that frame is destroyed, so are its dependent dialogs.
- When the frame is iconified, its dependent dialogs disappear from the screen.
- When the frame is deiconified, its dependent dialogs return to the screen. The AWT automatically provides this behavior.

modal Dialogs

- A dialog can be *modal*.
- When a modal dialog is visible, it blocks user input to all other windows in the program.
- The JDialogs that JOptionPane creates are modal.
- To create a non-modal dialog, you must use the JDialog class directly.

Dialog Examples

- Lets take a look at the DialogDemo.java example in the sample 2 – dialog folder
- This sample demonstrates many of the different types of dialogs that we can use in Java

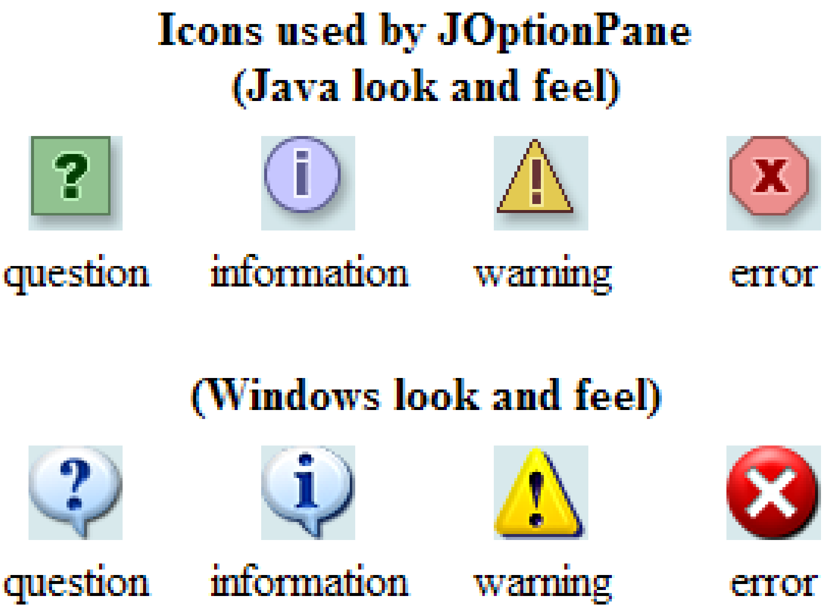
JOptionPane Features

- Using JOptionPane, you can create and customize several different kinds of dialogs.
- JOptionPane provides support for laying out standard dialogs, providing icons, specifying the dialog's title and text, and customizing the button text.
- Other features allow you to customize the components the dialog displays and specify where the dialog should appear onscreen.
- You can even specify that an option pane put itself into an internal frame (JInternalFrame) instead of a JDialog.

JOptionPane Features

- JOptionPane's icon support lets you easily specify which icon the dialog displays.
- You can use a custom icon, no icon at all, or any one of four standard JOptionPane icons
  - Question
  - Information
  - Warning
  - Error

- Each look and feel has its own versions of the four standard icons. The following figure shows the icons used in the Java look and feel.



Creating and Showing Simple Dialogs

- For most simple modal dialogs, you create and show the dialog using one of JOptionPane's show*Xxx*Dialog methods.

- If your dialog should be an internal frame, then add Internal after show — for example, showMessageDialog changes to showInternalMessageDialog.

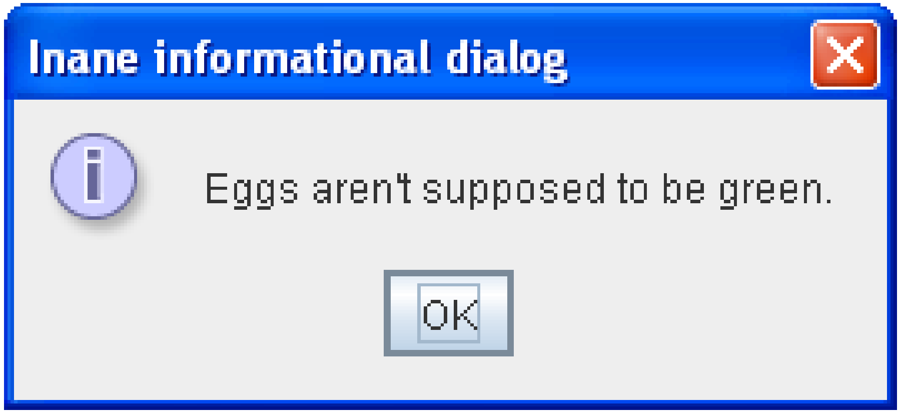
- The two most useful show*Xxx*Dialog methods are showMessageDialog and showOptionDialog

- The showMessageDialog method displays a simple, one-button dialog.

- The showOptionDialog method displays a customized dialog — it can display a variety of buttons with customized button text, and can contain a standard text message or a collection of components.

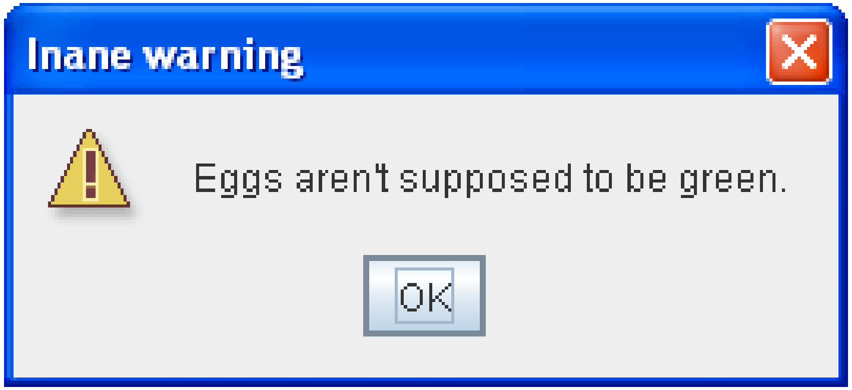
showMessageDialog

Lets take a look at some examples


Inane informational dialog

Eggs aren't supposed to be green.

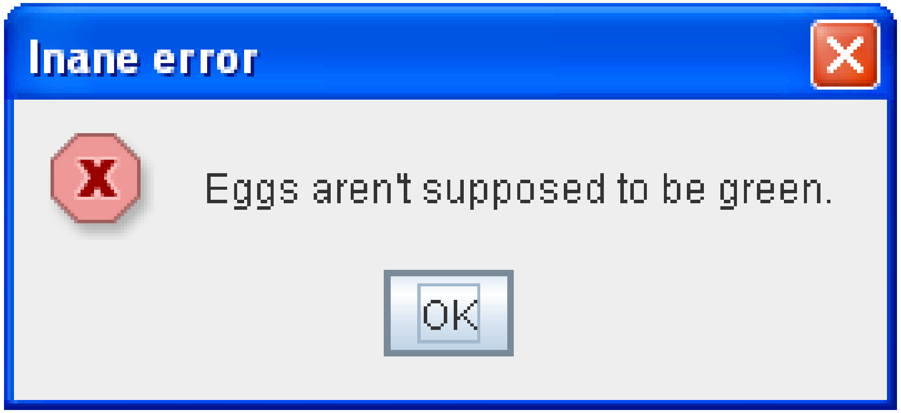
```
JOptionPane.showMessageDialog(frame, "Eggs aren't supposed to be green.");
```



Inane warning

Eggs aren't supposed to be green.

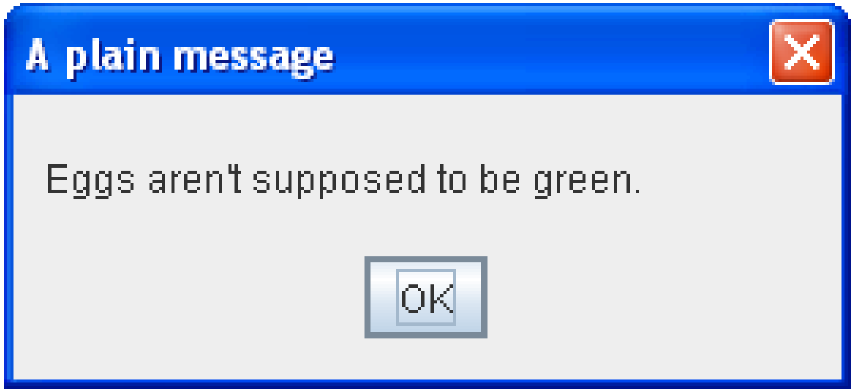
```
JOptionPane.showMessageDialog(frame, "Eggs aren't supposed to be green.",
"Inane warning", JOptionPane.WARNING_MESSAGE);
```



Inane error

Eggs aren't supposed to be green.

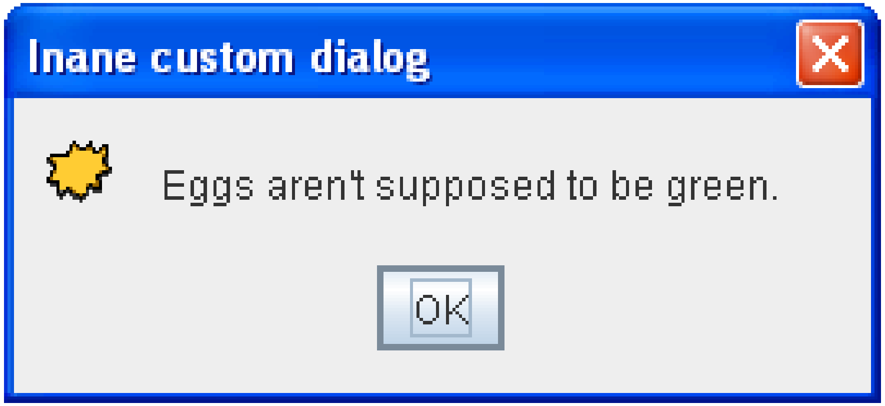
```
JOptionPane.showMessageDialog(frame, "Eggs aren't supposed to be green.",
        "Inane error", JOptionPane.ERROR_MESSAGE);
```



A plain message

Eggs aren't supposed to be green.

```
JOptionPane.showMessageDialog(frame, "Eggs aren't supposed to be green.",
"A plain message", JOptionPane.PLAIN_MESSAGE);
```



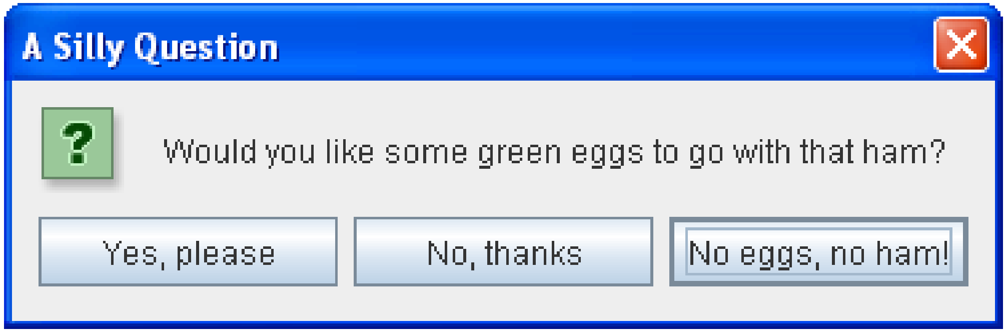
Inane custom dialog

Eggs aren't supposed to be green.

```
JOptionPane.showMessageDialog(frame, "Eggs aren't supposed to be green.",
    "Inane custom dialog", JOptionPane.INFORMATION_MESSAGE, icon);
```

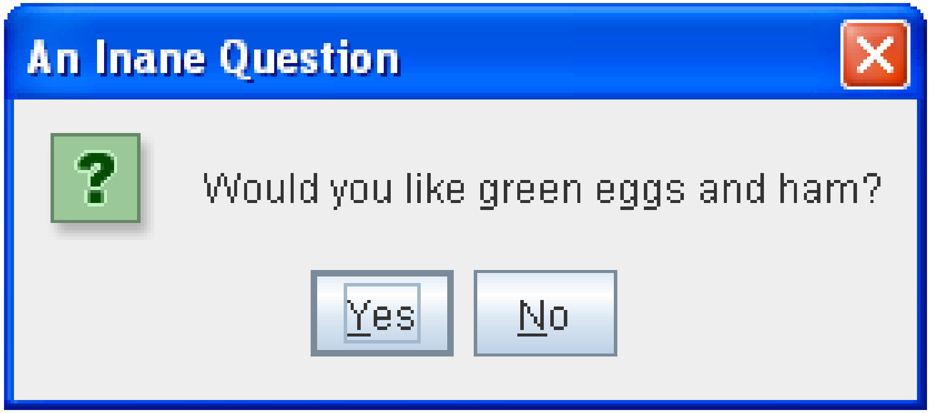
showOptionDialog

Lets take a look at some examples



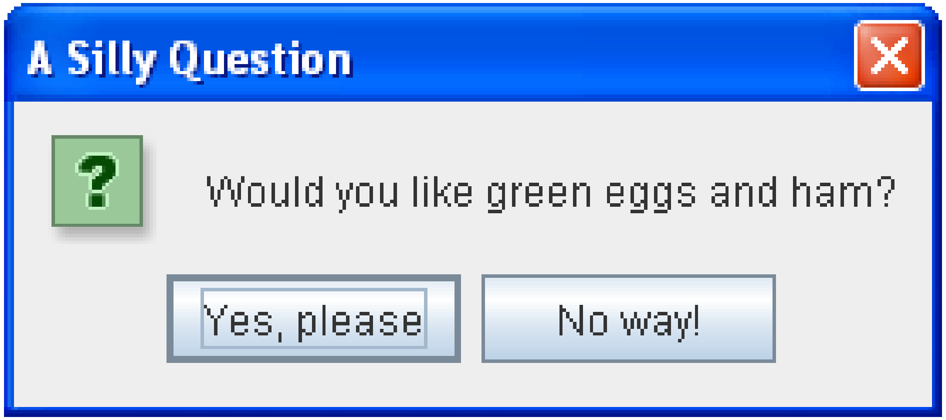
Would you like some green eggs to go with that ham?

```
//Custom button text

Object[] options = {"Yes, please", "No, thanks", "No eggs, no
ham!"};

int n = JOptionPane.showOptionDialog(frame, "Would you like
some green eggs to go " + "with that ham?", "A Silly Question",
JOptionPane.YES_NO_CANCEL_OPTION,
JOptionPane.QUESTION_MESSAGE, null, options, options[2]);
```


An Inane Question

Would you like green eggs and ham?

//default icon, custom title
int n = JOptionPane.showConfirmDialog(frame, "Would you like green eggs and ham?", "An Inane Question", JOptionPane.YES_NO_OPTION);



A Silly Question

Would you like green eggs and ham?

```
Object[] options = {"Yes, please", "No way!"};

int n = JOptionPane.showOptionDialog(frame, "Would you like green eggs
and ham?", "A Silly Question", JOptionPane.YES_NO_OPTION,
JOptionPane.QUESTION_MESSAGE, null, //don't use a custom Icon
options, //the titles of buttons options[0]); //default button title
```

Return Values

- As the previous code snippets showed, the showMessageDialog, showConfirmDialog, and showOptionDialog methods return an integer indicating the user's choice.

- The values for this integer are YES_OPTION, NO_OPTION, CANCEL_OPTION, OK_OPTION, and CLOSED_OPTION.

- Except for CLOSED_OPTION, each option corresponds to the button the user pressed. When CLOSED_OPTION is returned, it indicates that the user closed the dialog window explicitly, rather than by choosing a button inside the option pane.

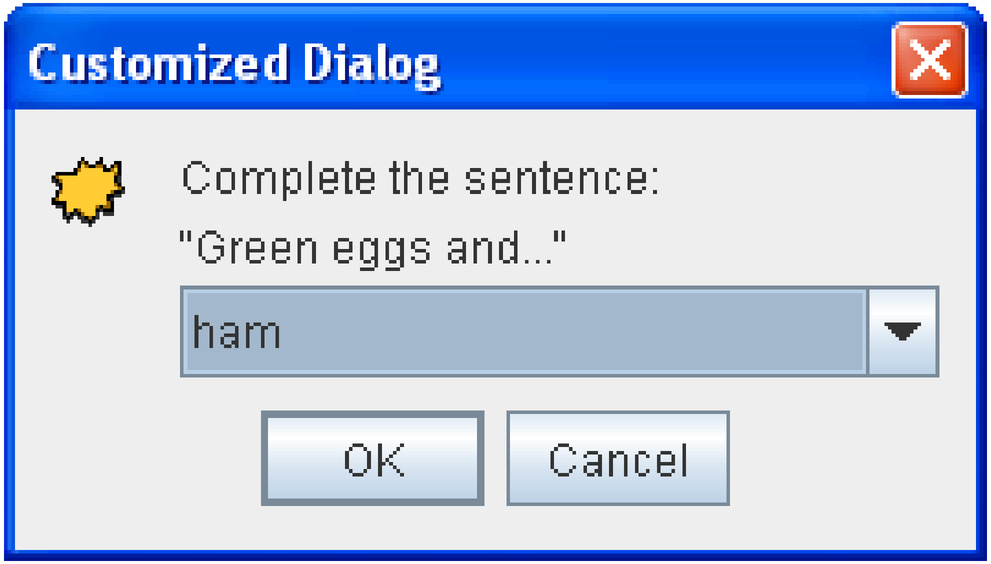
- Even if you change the strings that the standard dialog buttons display, the return value is still one of the pre-defined integers. For example, a YES_NO_OPTION dialog always returns one of the following values: YES_OPTION, NO_OPTION, or CLOSED_OPTION.

showInputDialog

Lets take a look at some examples

- The only form of show*Xxx*Dialog that doesn't return an integer is showInputDialog, which returns an Object instead.

- This Object is generally a String reflecting the user's choice.

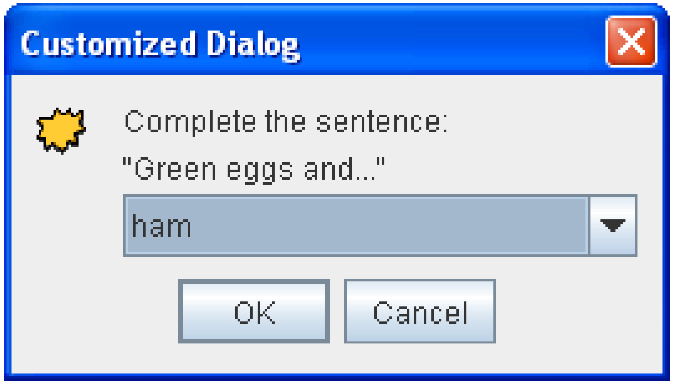


Complete the sentence:

"Green eggs and..."

ham

```
Object[] possibilities = {"ham", "spam", "yam"};
String s = (String)JOptionPane.showInputDialog(
            frame,
            "Complete the sentence:\n"
            + "\"Green eggs and...\"",
            "Customized Dialog",
            JOptionPane.PLAIN_MESSAGE,
            icon,
            possibilities,
            "ham");

//If a string was returned, say so.
if ((s != null) && (s.length() > 0)) {
setLabel("Green eggs and... " + s + "!"); return;
}
//If you're here, the return value was null/empty.
setLabel("Come on, finish the sentence!");
```



Customized Dialog

Complete the sentence:

"Green eggs and..."

ham

Labwork 9:

Create three sample JFrames. Demonstrate how to create the following using the three JFrames:

1. A simple message dialog
2. A options dialog with at least OK and Cancel
3. An input dialog to input some information and output to the screen what was selected