Introduction to JOptionPane

`JOptionPane` is a class in the `javax.swing` package that provides standard dialog boxes such as message, input, and confirmation dialogs. These dialogs are commonly used in GUI applications to interact with users.

Types of Dialogs:

Message Dialog: Displays information to the user. **Input Dialog**: Prompts the user for input. **Confirm Dialog**: Asks the user to confirm an action (e.g., Yes/No/Cancel). **Option Dialog**: Allows for more customized options.

**Displaying Message with `JOptionPane.showMessageDialog`

```
import javax.swing.JOptionPane;
public class Main {
    public static void main(String[] args) {
        JOptionPane.showMessageDialog(null, "This is a simple message.");
    }
}
```

Customizing the Message Dialog: Title: You can add a custom title to the dialog. Message Type: You can specify the type of message, such as `INFORMATION_MESSAGE`, `ERROR_MESSAGE`, etc.

```
JOptionPane.showMessageDialog(null, "This is an informational message.",
"Information", JOptionPane.INFORMATION_MESSAGE);
```

Using a Custom Icon 🧮

```
import javax.swing.ImageIcon;

public class Main {
    public static void main(String[] args) {
        ImageIcon icon = new ImageIcon("path_to_icon.png");
        JOptionPane.showMessageDialog(null, "Message with a custom icon", "Custom
Icon", JOptionPane.INFORMATION_MESSAGE, icon);
    }
}
```

Getting User Input

- Basic User Input

```
String name = JOptionPane.showInputDialog("Enter your name:");
System.out.println("User's name: " + name);
```

- Input Dialog with a custom message and initial value

```
String age = JOptionPane.showInputDialog(null, "Enter your age:", "Age Input",
JOptionPane.QUESTION_MESSAGE, null, null, "18").toString();
System.out.println("User's age: " + age);
```

- Input Dialog with a Drop-Down Selection 🧮

```
String[] options = {"Option 1", "Option 2", "Option 3"};
String selection = (String) JOptionPane.showInputDialog(null, "Choose an option:",
"Options", JOptionPane.QUESTION_MESSAGE, null, options, options[0]);
System.out.println("User selected: " + selection);
```

**Comfirming User Actions with `JOptionPane.showConfirmDialog`

- Basic Confirm Dialog

```
int response = JOptionPane.showConfirmDialog(null, "Do you want to proceed?",
"Confirm", JOptionPane.YES_NO_OPTION);
if (response == JOptionPane.YES_OPTION) {
    System.out.println("User chose Yes.");
} else {
    System.out.println("User chose No.");
}
```

- Custom Confirm Dialog

```
int response = JOptionPane.showConfirmDialog(null, "Do you want to save changes?",
"Save", JOptionPane.YES_NO_CANCEL_OPTION, JOptionPane.WARNING_MESSAGE);
```

**Advanced Option Dialog with `JOptionPane.showOptionDialog` The `showOptionDialog` method provides full control over the dialog box.

```
Object[] options = {"Save", "Don't Save", "Cancel"};
int choice = JOptionPane.showOptionDialog(null, "Do you want to save changes?",
"Save Confirmation", JOptionPane.YES_NO_CANCEL_OPTION,
JOptionPane.QUESTION_MESSAGE, null, options, options[0]);
```

**Using HTML to format Text in `JOptionPane`. You can use HTML to format the text inside the `JOptionPane` dialog boxes.

- Basic HTML example

```java
JOptionPane.showMessageDialog(null, "<html><h1>Formatted Message</h1><p>This message is <b>bold</b> and this is <i>italic</i>.</p></html>");
```

- Example with Input Dialog 🧮

```java
String name = JOptionPane.showInputDialog(null, "<html><h2>Enter your <font color='blue'>name</font>:</h2></html>");
System.out.println("User's name: " + name);
```

- Example with Confirm Dialog

```java
int response = JOptionPane.showConfirmDialog(null, "<html><p>Do you want to <b>proceed</b> with this action?</p></html>", "Confirmation", JOptionPane.YES_NO_OPTION);
```

- Example with Custom Icon and HTML

```java
ImageIcon icon = new ImageIcon("path_to_icon.png");
JOptionPane.showMessageDialog(null, "<html><h3>This is a custom message with <font color='green'>HTML</font> formatting.</h3></html>", "Custom Dialog", JOptionPane.INFORMATION_MESSAGE, icon);
```