

# ADDING ACTION LISTENERS

## SETTING UP THE BASIC WINDOW

You can modify the simple application that you made in the previous tutorial, or you can create a new one from scratch.

The screenshot shows an IDE window with the following code:

```
1 package javaapplication1;
2
3
4 import javafx.application.Application;
5 import javafx.geometry.Pos;
6 import javafx.scene.Scene;
7 import javafx.scene.control.Button;
8 import javafx.scene.control.Label;
9 import javafx.scene.layout.VBox;
10 import javafx.stage.Stage;
11
12 /**
13  * @author Yatish
14  */
15 public class JavaApplication1 extends Application{
16
17     public static void main(String[] args) {
18         launch(args);
19     }
20
21     @Override
22     public void start(Stage primaryStage) {
23         VBox box = new VBox();
24         box.setAlignment(Pos.CENTER);
25
26         Scene myScene = new Scene(box, 800, 600);
27
28         Label lblText = new Label();
29         lblText.setText("Hello World");
30
31         Button btn = new Button();
32         btn.setText("Say 'Hello World'");
33
34         box.getChildren().add(lblText);
35         box.getChildren().add(btn);
36
37         primaryStage.setTitle("Hello World");
38         primaryStage.setScene(myScene);
39         primaryStage.show();
40     }
41 }
42
43 }
```

Callouts in the image provide the following explanations:

- New imports:** Points to the import statements from line 4 to line 10.
- This time the layout we want to experiment with is a VBox.** Points to the `VBox box = new VBox();` line (23).
- Create a button and set the text on it.** Points to the `Button btn = new Button();` and `btn.setText("Say 'Hello World');` lines (31-32).
- Add button to VBox layout.** Points to the `box.getChildren().add(btn);` line (35).

If you run the application, You should see the new button below the original “Hello world” label. However, nothing happens yet when you click the button.

From [yatishparmar.com](http://yatishparmar.com)

## ADDING AN ACTION LISTENER

```
import javafx.event.ActionEvent;  
import javafx.event.EventHandler;
```

We need to add two more imports to make the action listener work.

You can now add in the action listener for the button. The logical place to add it in your code is below where it is created and configured.

```
btn.setOnAction(new EventHandler<ActionEvent>() {  
  
    @Override  
    public void handle(ActionEvent event) {  
        System.out.println("Hello World!");  
    }  
});
```

```
btn.setOnAction(new EventHandler<ActionEvent>() {
```

This creates an action listener. It defines a new anonymous inner class that implements EventHandler. An anonymous inner class is a class within a method that doesn't have a name.

```
@Override  
public void handle(ActionEvent event) {  
    System.out.println("Hello World!");  
}
```

EventHandler is an interface. This means it doesn't do anything, it provides a signature methods without implementation. It is like structure for programmers to follow when they are creating their own application. Here we override the empty handle method in EventHandler with our own functionality.

Run the application and click on the button. You should see a message in the output window in your IDE.

See if you can extend the application to output a message by adding a new label in the VBox when the button is clicked.