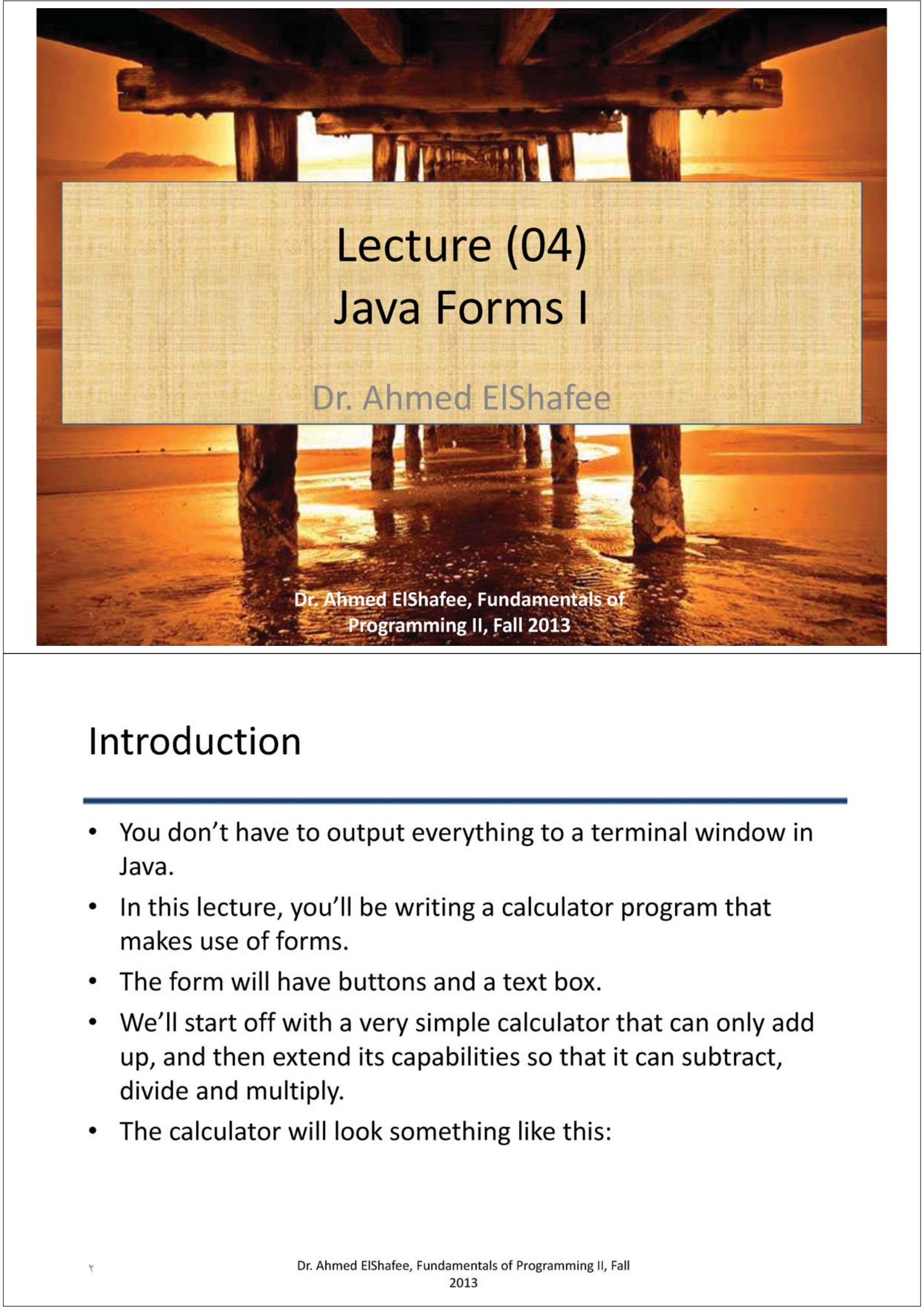




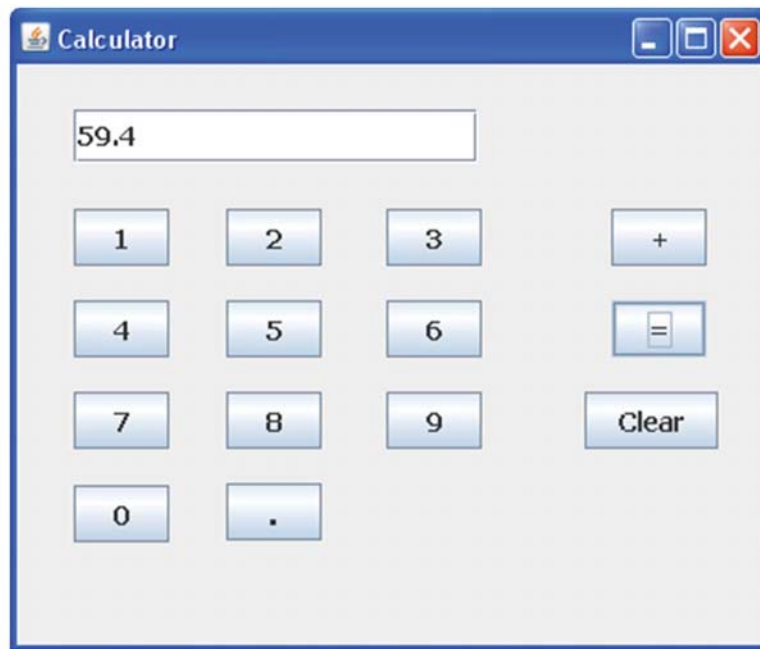
Lecture (04) Java Forms I

Dr. Ahmed ElShafee

Dr. Ahmed ElShafee, Fundamentals of
Programming II, Fall 2013

Introduction

- You don't have to output everything to a terminal window in Java.
- In this lecture, you'll be writing a calculator program that makes use of forms.
- The form will have buttons and a text box.
- We'll start off with a very simple calculator that can only add up, and then extend its capabilities so that it can subtract, divide and multiply.
- The calculator will look something like this:



Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

NetBeans and Java Forms

- The NetBeans development environment, has greatly simplified the creation of forms, and we'll use this to drag and drop controls onto a frame.
- Rather than go through reams of GUI theory, we'll just jump straight into it.

-
- Create a new project for this by clicking **File > New Project** from the NetBeans menu at the top.
 - Select **Java > Java Application** from the list boxes, and then click the **Next button**.

-
- On step 2 of the wizard, type MyCalculator as the project name. At the bottom, uncheck “Create main class”.
 - This is because a main method will be created for us by NetBeans when we add a form.
 - Step 2 of the wizard should look like this:

Name and Location

Project Name:

Project Location:

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

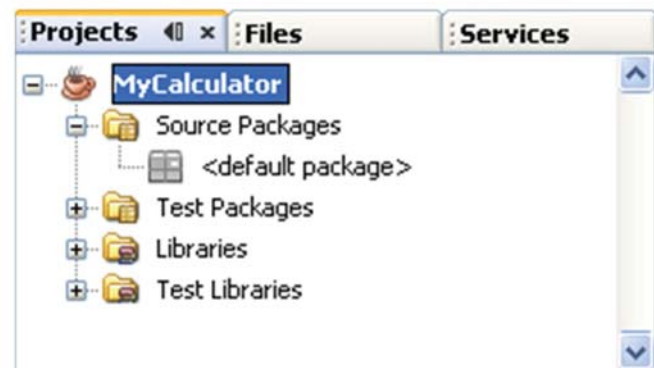
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☐ Create Main Class

☒ Set as Main Project

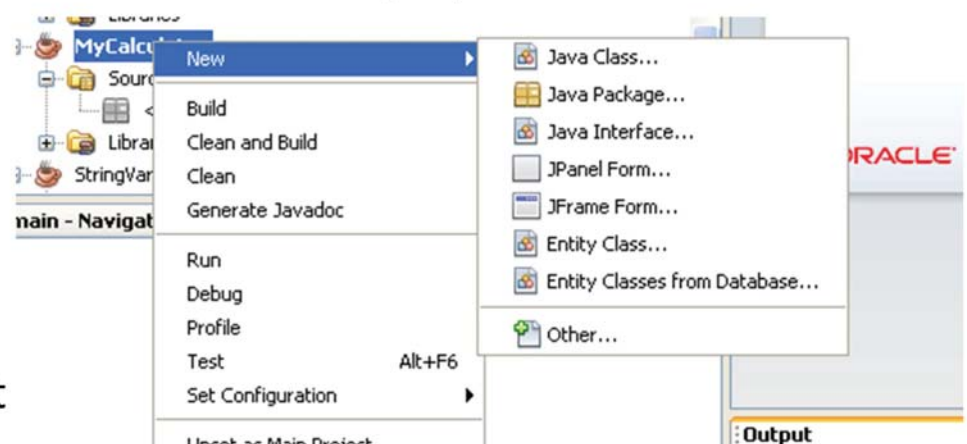
- Click the **Finish** button, and NetBeans will create the project, but not much else.
- Have a look at the Projects area on the left in NetBeans and you should see this (If you can't see the Projects area, click **Window > Projects** from the NetBeans menu at the top)



y

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

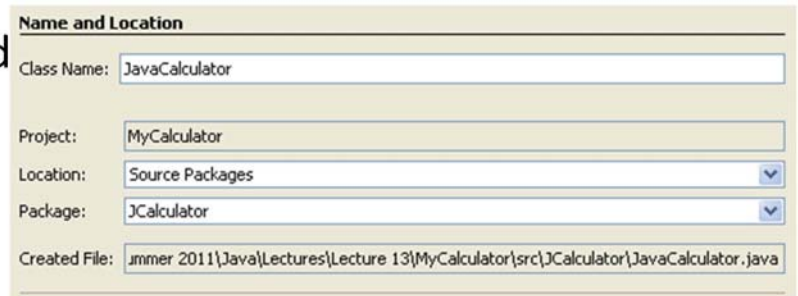
- Normally, there's a java file under the Source Packages name. But because we unchecked the "Create main class" box, there's no java class file there.
- What we'll do is to add a Form to the project.
- When the form is created, it will be created in its own java class file.
- To add a form, right click the project name in the Projects window.
- A menu will appear:



Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

Form

- Select **New > JFrame Form** from the menu.
- When you do, you should see the following wizard appear:
- Here, you are being asked for a name for your Class, and a package name.
- We've already created the project, and called it **MyCalculator**.



Name and Location

Class Name:

Project:

Location:

Package:

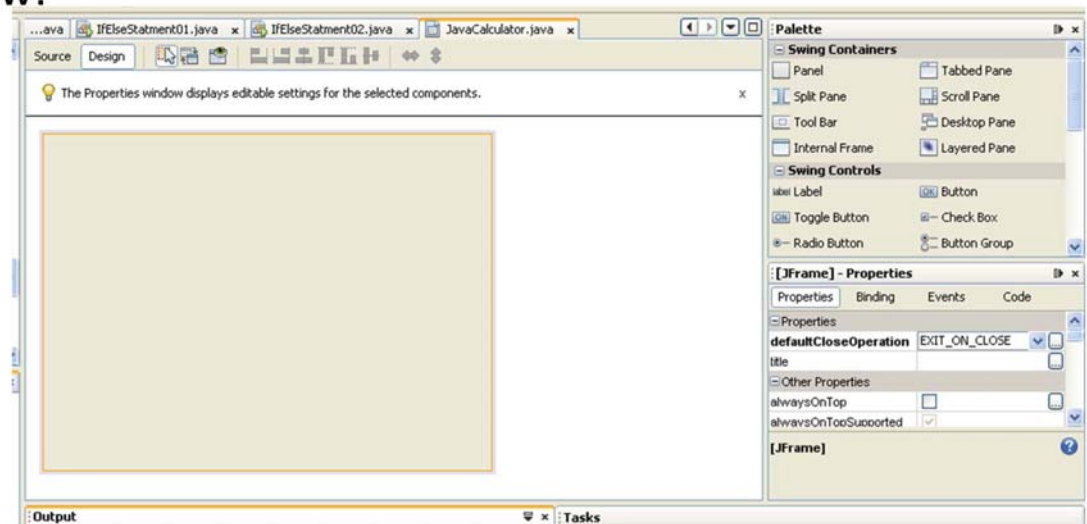
Created File:

- The package name and class will go into the project.
- So, for your Class Name type **JavaCalculator**.
- In the blank package text box, type **jCalculator**.

9

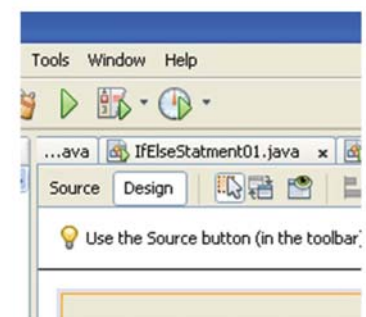
Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- So we're creating a class called **JavaCalculator**, which is in the **jCalculator** package, which is in the **MyCalculator** project
- Click the **Finish** button to complete the process.
- When the wizard is finished, it will create a blank form in the main window:



10

- The form is a blank, at the moment, and has an orange rectangle surrounding it.
- The orange rectangle means that the form is the currently selected object.
- Click away and you'll see a blue rectangle instead. This means that the form is not selected.
- Click back onto the form to select it, and you'll see the orange rectangle again.
- Note the two buttons at the top, **Source** and **Design**.
- You're in Design at the moment.
- Click on Source to see the following code:



Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- You can expand and contract the plus symbols to reveal and hide code.
- Notice, though, that a **main method** has been created.
- When the program starts it's the main method that will be called.
- It's creating a new object from our form, and setting its visible property to **true**.

```
package jCalculator;

/**...*/
public class JavaCalculator extends javax.swing.JFrame {

    /** Creates new form JavaCalculator */
    public JavaCalculator() {
        initComponents();
    }

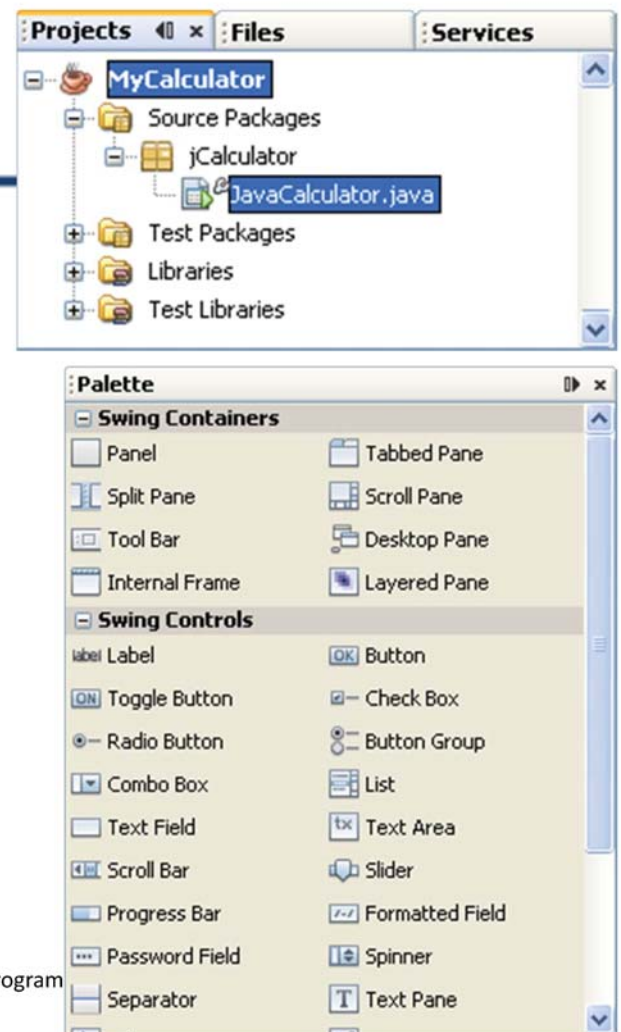
    /**...*/
    @SuppressWarnings("unchecked")
    Generated Code

    /**...*/
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new JavaCalculator().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    // End of variables declaration
}
```

Ahmed ElShafee, F

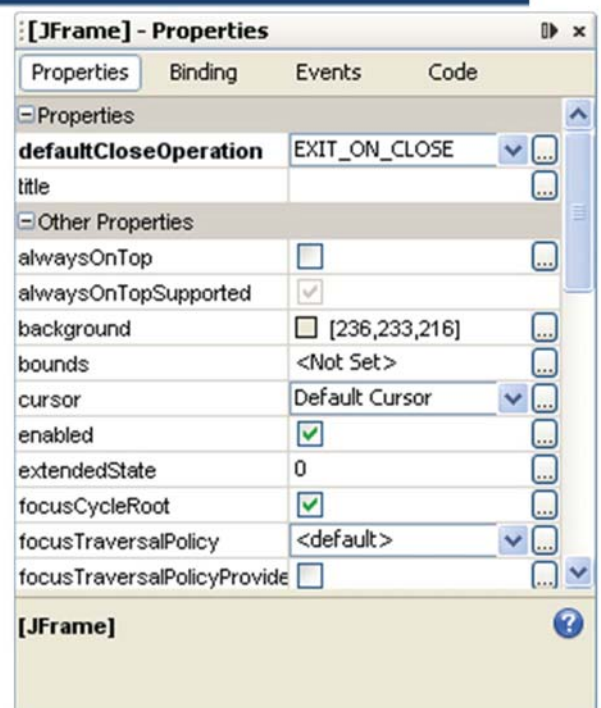
- Have a look at the Projects area on the left again. You'll see that a package and a
- class file have been added:
- Click back on the Design button at the top.
- You'll see your blank form again.
- To the right, you'll have noticed two areas: a Palette with a lot of controls in it, and a Properties area.
- The Palette should look like this:



١٣

Dr. Ahmed ElShafee, Fundamentals of Program
2013

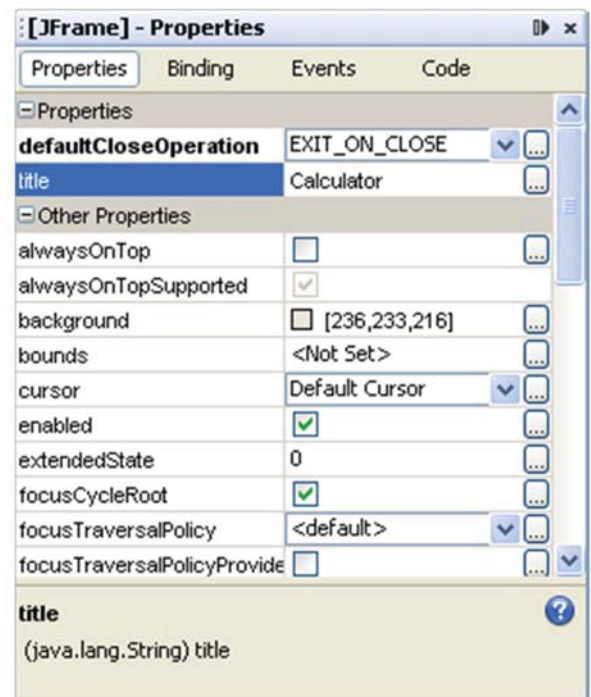
- And the Properties area should look like this:
- (If you can't see them, click **Windows > Palette** and **Windows > Properties** from the NetBeans menu.)



١٤

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

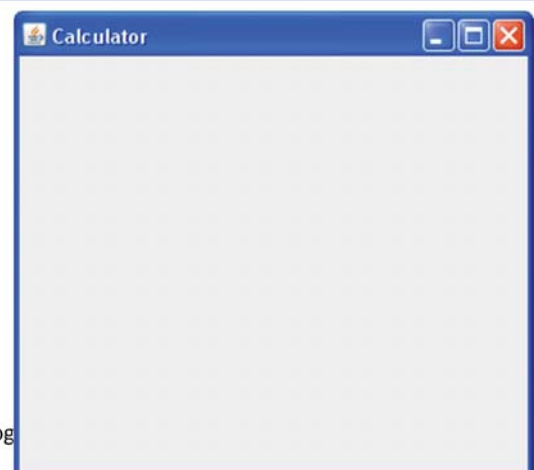
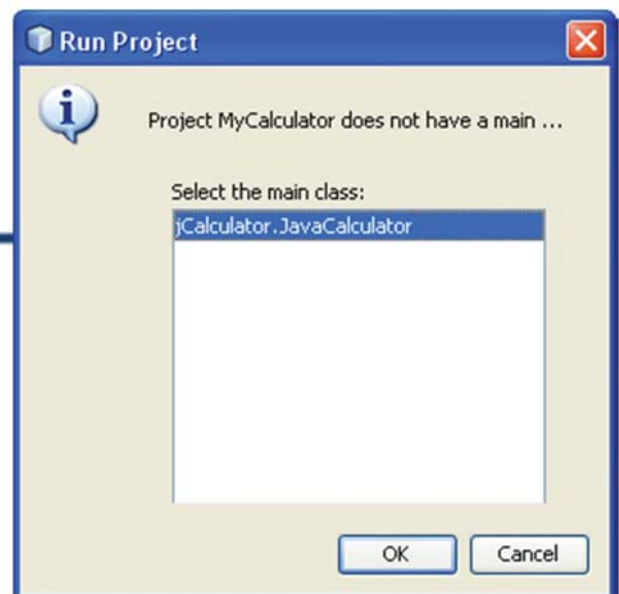
- A property of an object is a list of the things you can do with it, like set the background color, set the text, set the font, and lots more.
- Let's change the title.
- Make sure your form is selected. If it is, it will be surrounded with an orange rectangle.
- The properties area will say **JFrame** at the top.
- Click inside of the title area and type **Calculator**:



10

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

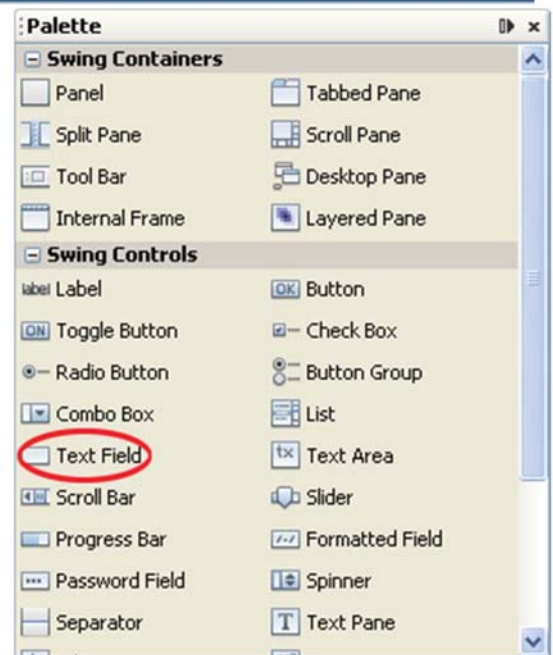
- Then press the enter key.
- To see what effect changing the title property has, run your program in the usual way.
- When you run the program for the first time, NetBeans will display a dialogue box asking you which main method you want to use:
- You only have one, which is in your **JavaCalculator class**, so just click OK.
- Your form should then appear.



Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

Components

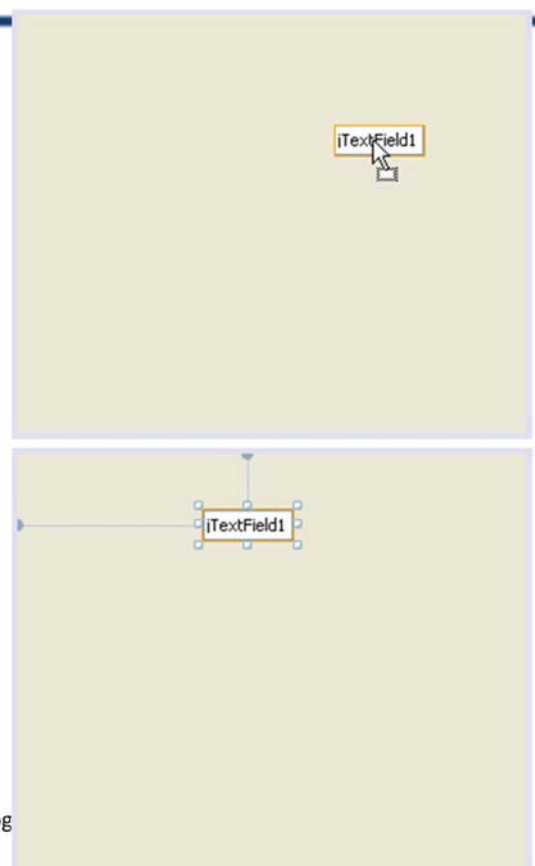
- The form has the title you just typed, plus some default icons for minimize, maximize and close.
- Click the X to close your program, and return to the Design environment.
- What we need now is a text box and some buttons.
- Let's add the text box first.
- Locate the Text Field control in the Palette:



١٧

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

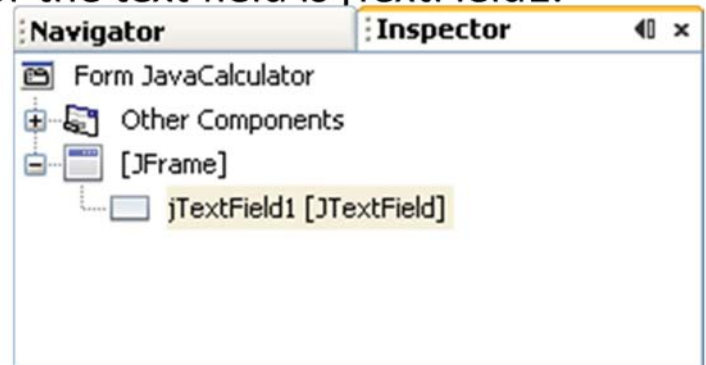
- Controls in the Palette can be dragged onto a form. So click on **Text Field** to select it.
- Now hold your left mouse button down on the Text Field.
- Keep it held down and drag the control onto the form:



١٨

Dr. Ahmed ElShafee, Fundamentals of Prog 2013

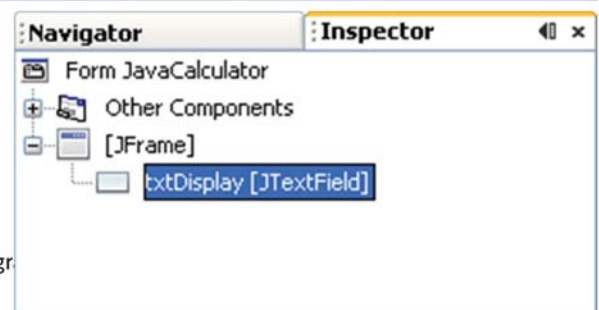
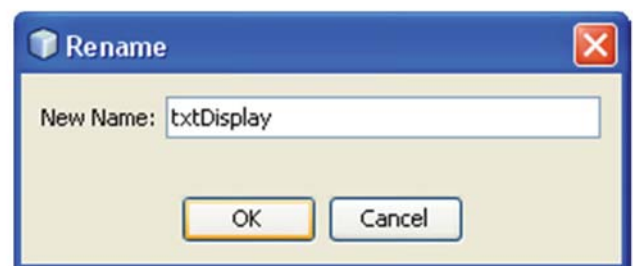
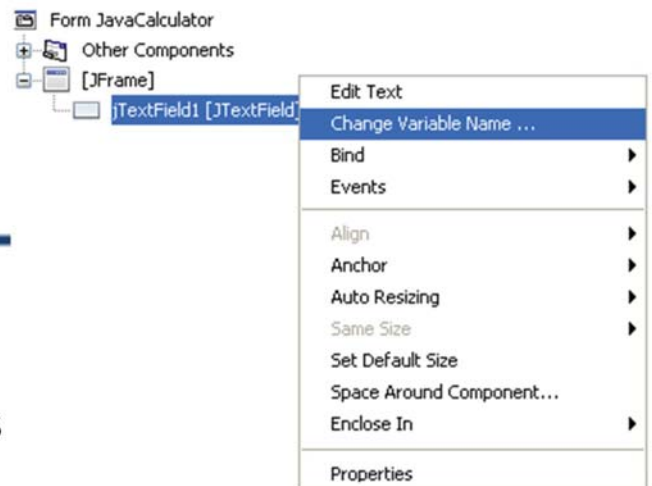
- The squares around the text field are sizing handles. You can hold down your left mouse button on one of the squares and drag to a new width or height.
- The dotted lines are position indicators, one for the left position and one for the top position.
- Notice that the default name for the text field is jTextField1. Let's change that.
- With the text field selected, have a look at the Inspector area in the bottom left: (If you can't see an Inspector area, click



Window > Navigating > Inspector from the NetBeans menu bar.)

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- As you can see, jTextField1 is selected in the Inspector.
- This area shows you what objects you have on your forms.
- You can also rename an object from here.
- To do so, right click on jTextField1. From the menu that appears, select **Change Variable Name**.
- When you click on Change Variable Name, a dialogue box appears.
- Type a new name for the Text Field. Call it **txtDisplay**.



Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- Now have a look at your code again by clicking the **Source** button in the main window.
- When your code appears, scroll down to the bottom.
- You'll see that a new private field variable has been added:

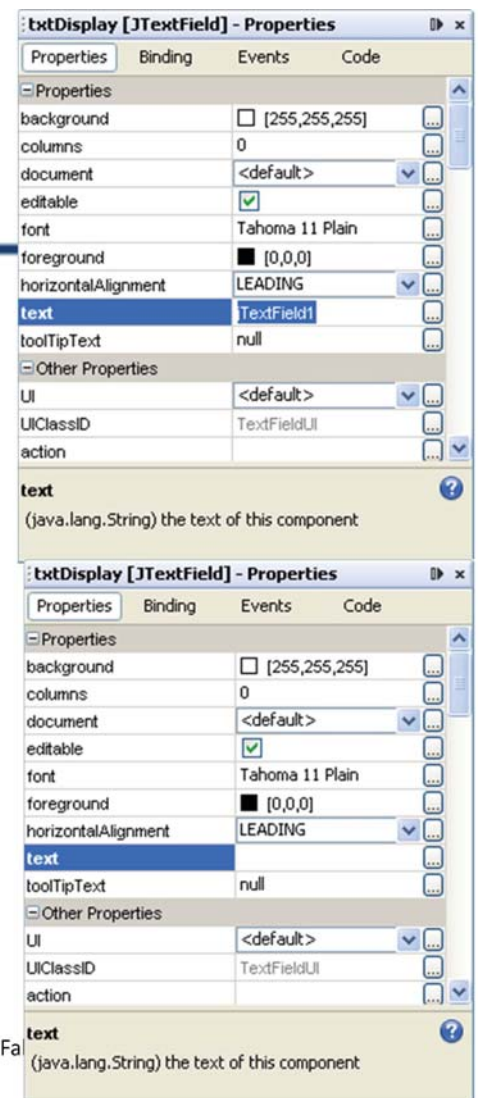
```
/**...*/
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new JavaCalculator().setVisible(true);
        }
    });
}
```

```
// Variables declaration - do not modify
private javax.swing.JTextField txtDisplay;
// End of variables declaration
```

٢١

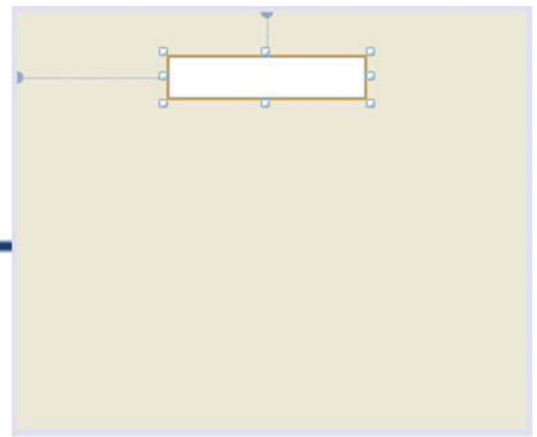
2013

- So a **JTextField** variable has been set up with the name **txtDisplay**.
- The “javax.swing” part is a reference to a set of packages that are used for GUI development.
- Click on the **Design** button at the top to return to your form.
- The text field has some default text in, at the moment.
- You can add your own text by changing the text property of the text field.



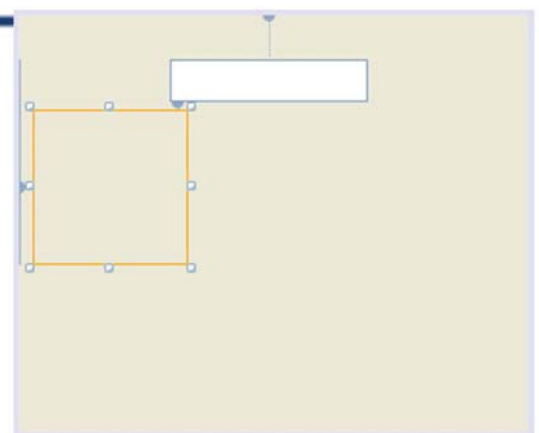
٢٢

- The text field on our calculator will obviously be used for the output of the calculation.
- You add a button to a form in the same way you do for text fields – drag and drop
- As we're going to be adding lots of buttons, it's a good idea to add the buttons to a control called a Panel.
- If you need to move the buttons, you can then just move the Panel instead.
- All the buttons will then move with the Panel.



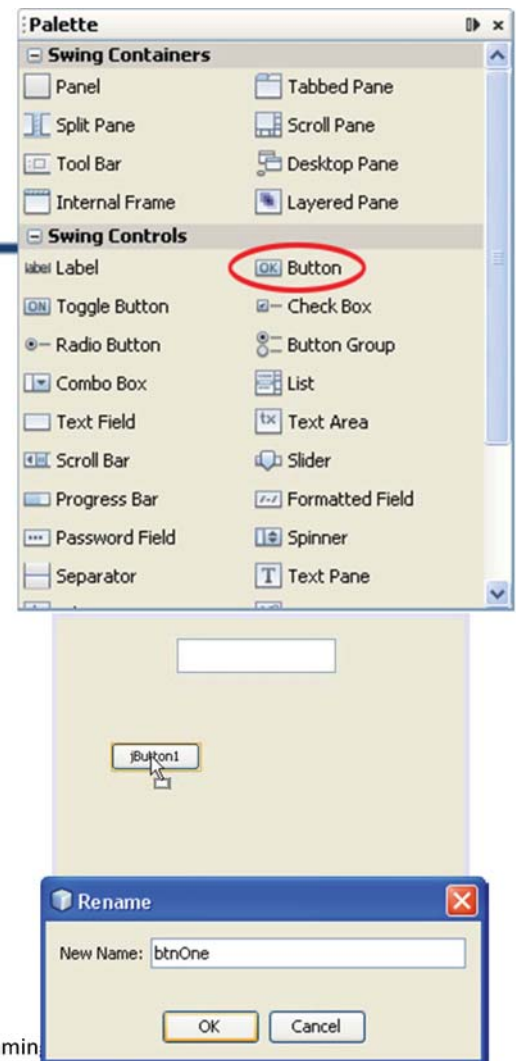
Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- Drag one on to your form. You can't see a Panel, as they'll have the same color as the form.
- But you can select it:
- Drag the sizing handles so that the Panel fills most of the form



Dr. Ahmed ElShafee, Fundamentals of Programm 2013

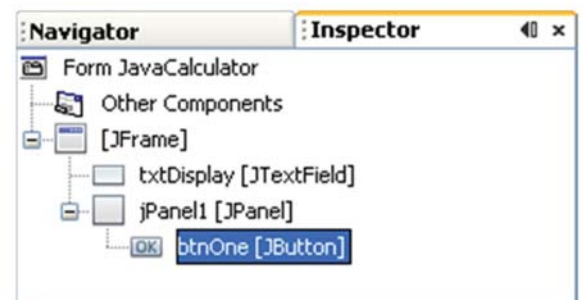
- We can now add a button.
- Locate the Button control in the Palette:
- Drag one on to your Panel control:
- The default name of the button is jButton1. Change this name just like you did for the text field: right click in the Inspector area, and select **Change variable name**.
- Change the name of the button to **btnOne**:



٢٥

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- The name of the button will have changed in the Inspector:
- A new line of code has also been added:
- The new variable name is **btnOne**, and it is a **JButton** object, which is a Swing control.
- Note, too, that a variable has been set up for the panel (we've left this on the default name).

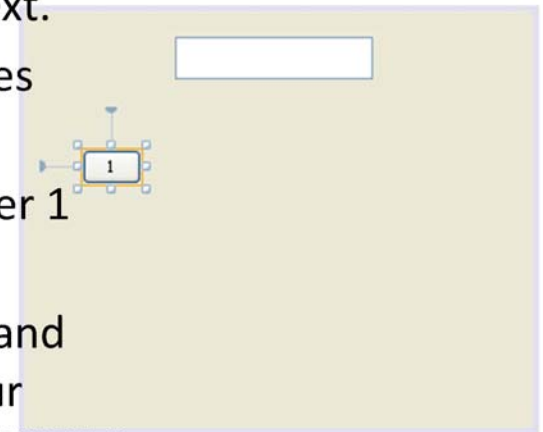
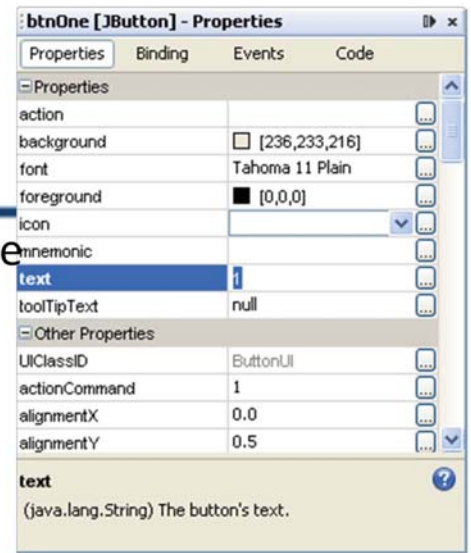


```
// Variables declaration - do not modify
private javax.swing.JButton btnOne;
private javax.swing.JPanel jPanel1;
private javax.swing.JTextField txtDisplay;
// End of variables declaration
```

٢٦

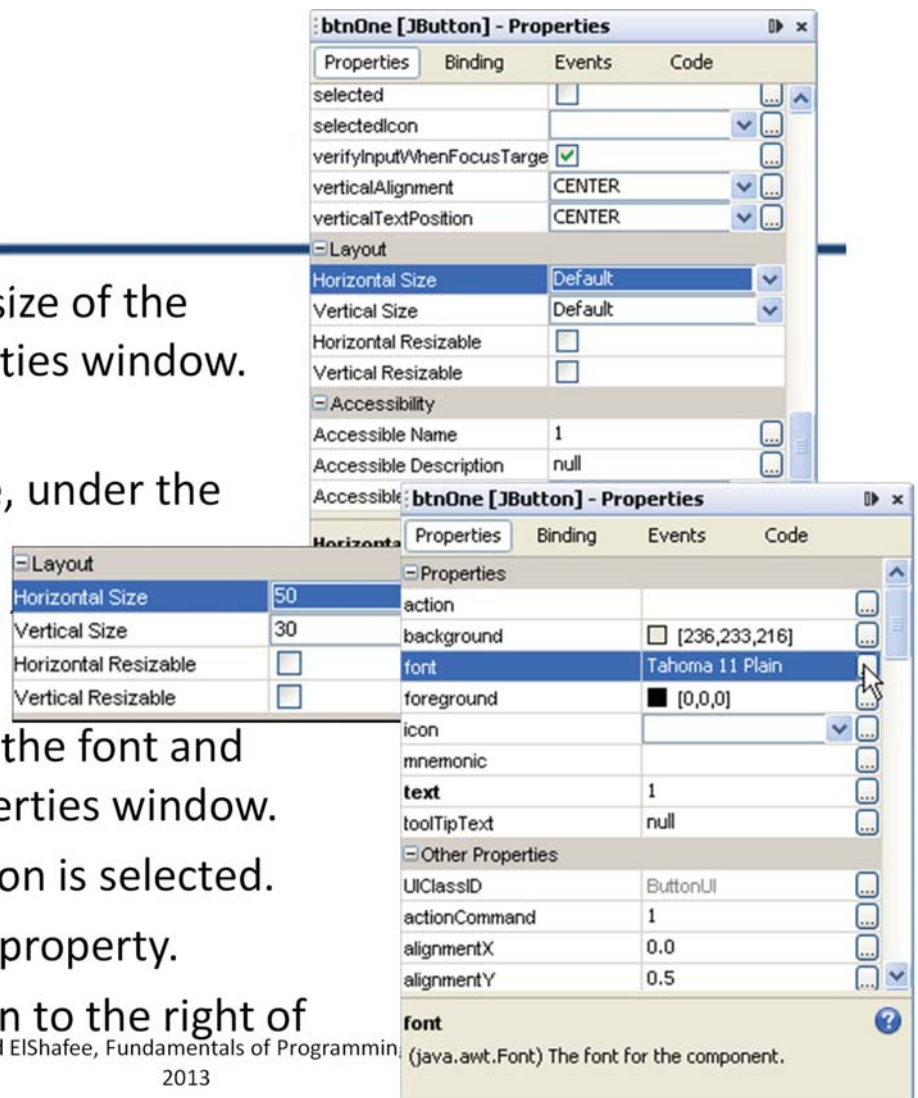
Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- Go back to Design view and make sure the button is selected.
- We can change the text on the button.
- What we want is a number, one number for each button on the calculator.
- The text property is used for button text.
- So locate this property in the properties window.
- Delete the default and type the number 1 instead:
- Press the enter key on your keyboard and you should see the text change on your button. It will also be resized.



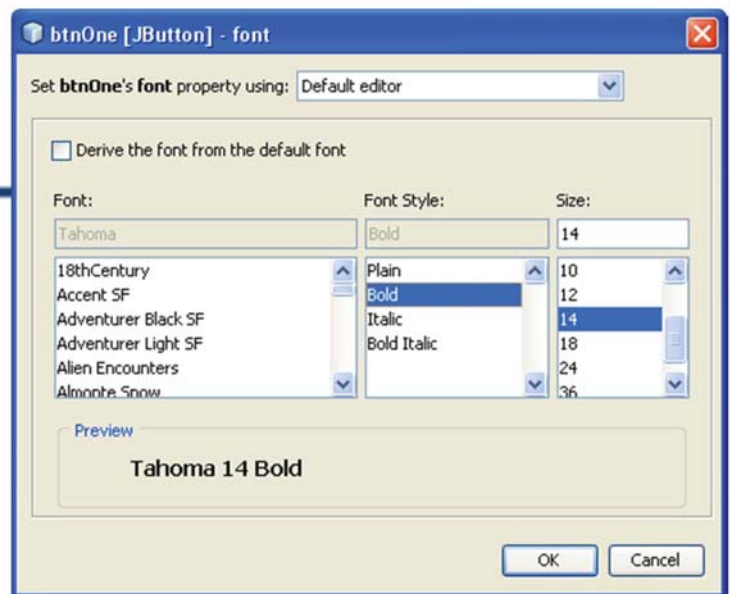
Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- You can change the size of the button in the properties window. Locate Horizontal
- Size and Vertical Size, under the Layout heading:
- Change these values default to 50, 30:
- You can also change the font and font size in the properties window.
- Make sure your button is selected.
- Now locate the font property.
- Click the small button to the right of the font row:



Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

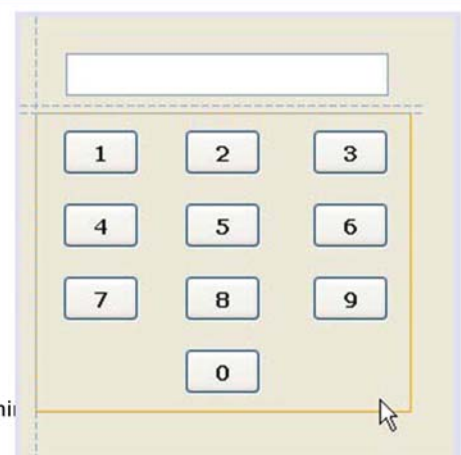
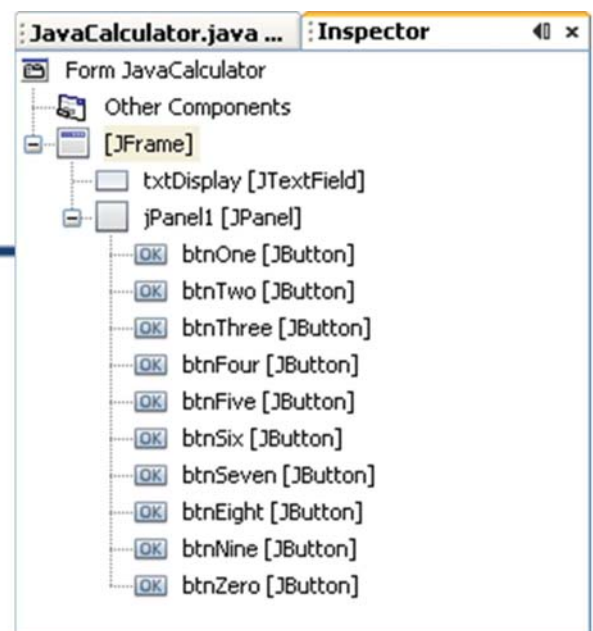
- When you click the font button, you should see a dialogue box appear.
- Change the font size to 14 points, and make it bold:
- You now need to add nine more buttons in the same way, for the numbers 2 to 9, and a 0.
- Change each variable name to btnTwo, btnThree, btnFour, etc.



٢٩

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

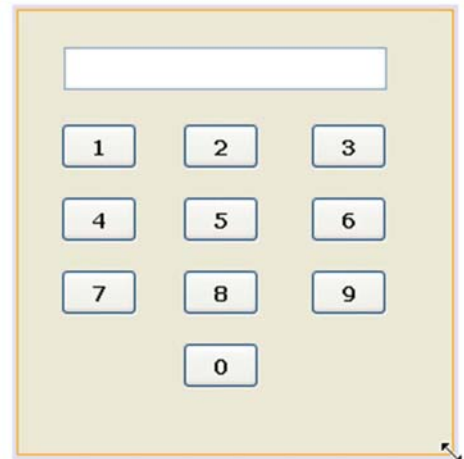
- Delete the default text, and enter a number instead.
- Then change the size of each button to 50, 30.
- Finally, change the font property of each button.



٣٠

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

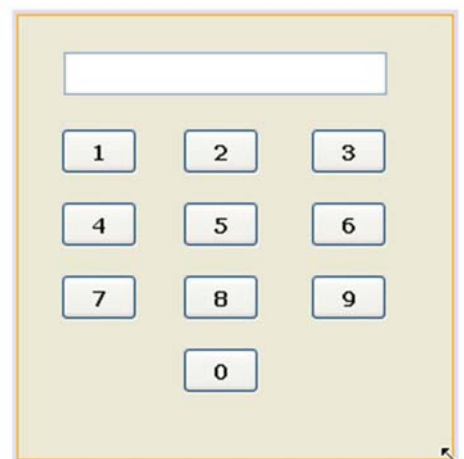
- Only three more buttons to add: a plus button, an equals button, and a clear button.
- our calculator doesn't have any room on the right.
- But you can easily resize a form.
- Click on just the form, and not the panel or the text field.
- If you do it right, you should see an orange rectangle surrounding the form.



٣١

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

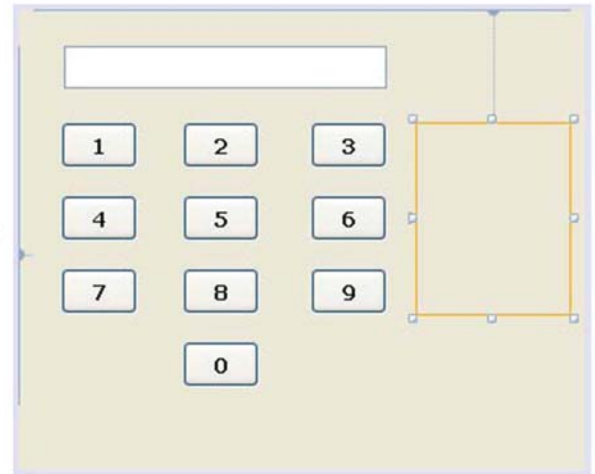
- Now move your mouse to the edges of the form.
- The mouse pointer should change shape, as in the image below (bottom right):



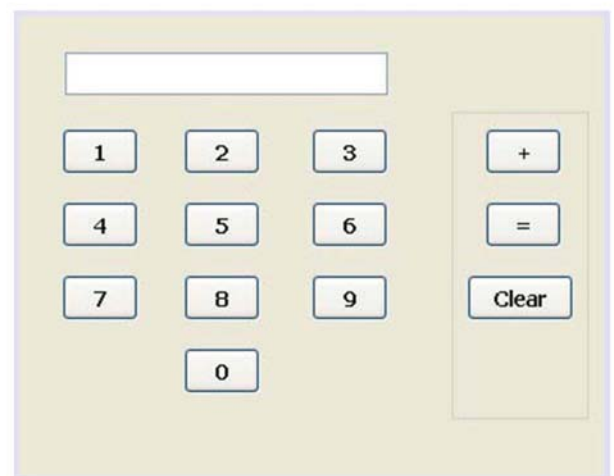
٣٢

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

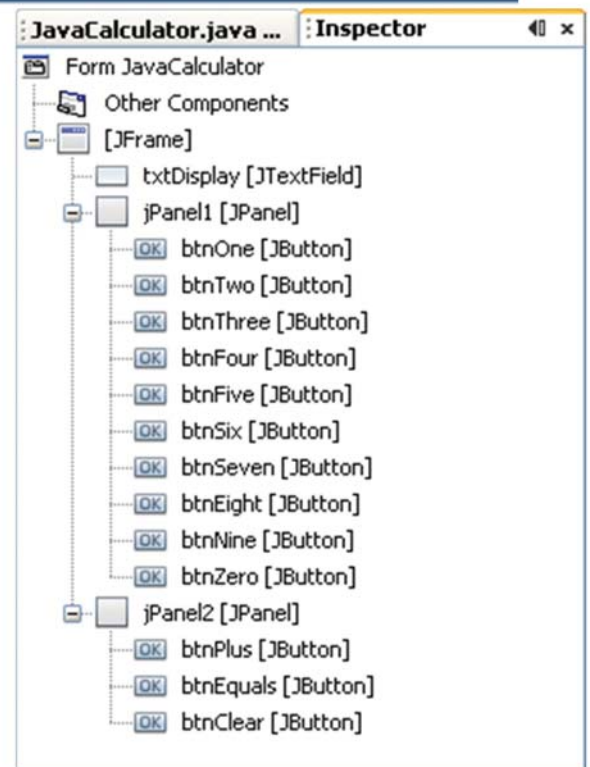
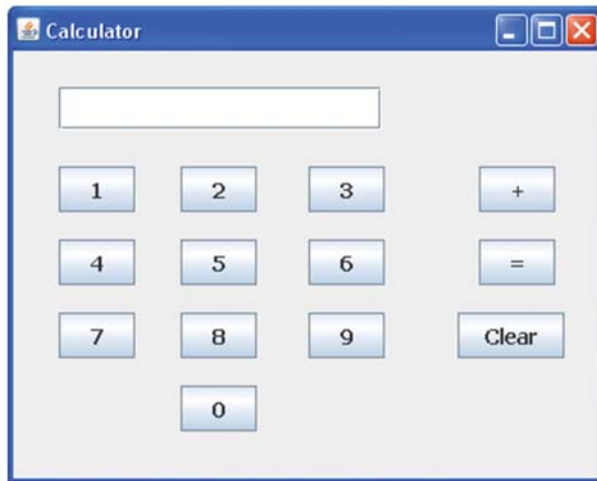
- Add a panel in the space you've just created:
- Add three buttons to the new panel. Change the variables names to: btnPlus, btnEquals, and btnClear.
- For the text for the buttons, type a + symbol for the Plus button, a = symbol for equals button, and the word "clear" for the clear button.



- Change the font to the same as the number buttons, 14 point bold. The size for the
- plus and equals button should be the same as the number buttons: 50, 30. For the
- Clear button, change it to 70, 30. Your form should then look like this:



- The Inspector area of NetBeans should look like this:
- You can run your form at this stage, just to see what it looks like:



٣٥

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

Events

- In programming terms, an event is when something special happens.
- For forms, this means things like buttons being clicked, the mouse moving, text being entered into text fields, the programming closing, and a whole lot more.

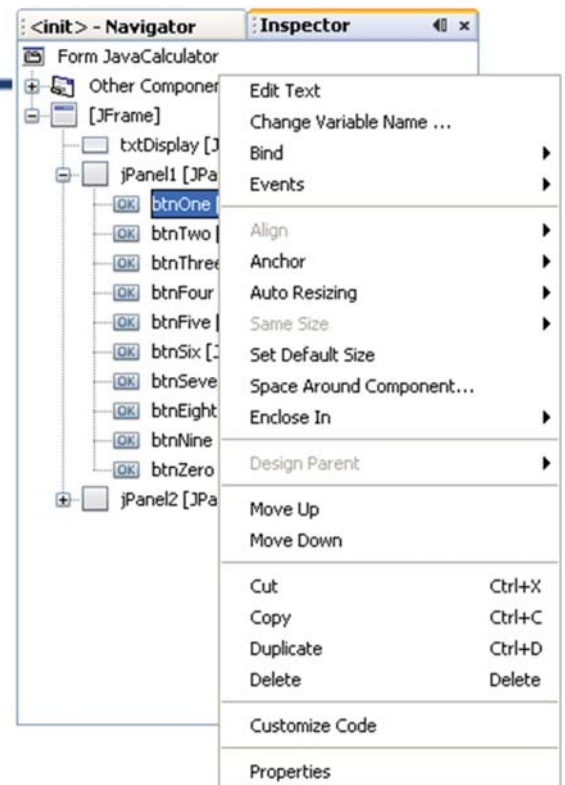
٣٦

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

-
- If we want to add $3 + 2$, we first need to click the 3 button.
 - The number 3 will then appear in the text field.
 - The plus button is clicked next, and this alerts the program to the fact that we want to add things.
 - It will also clear the text field ready for the next number.
 - The next number is 2, and we store this value along with the 3.
 - The equals button is clicked to get the total, and this is where we take the stored numbers (3 and 2) and add them together.
 - Finally, the answer is stored in the text field.

-
- The first problem is how to get at the numbers on the buttons.
 - We can do this by returning the text property of the button.
 - Once we have the text, we can put it into the text box.
 - But we need an `ActionEvent` for when the button is clicked.

- In Design view in NetBeans, select your number 1 button.
- Now have a look at the Inspector window in the bottom left.
- Locate your **btnOne** button.
- Now right click to see the following menu appear:
- Select **Events** from the menu.
- From the submenu, click on **Action**, and then **actionPerformed**:



٣٩

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

- Or simply double click the button



٤٠

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall 2013

-
- When you click on **actionPerformed**, you will create a code stub for **btnOne**:

```
private void  
btnOneActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
}
```

- The first line is just a method with an **ActionEvent** object between the round brackets.
- When the button is clicked, whatever code we write between the curly brackets will get executed

setText

- if wanted to put something into the text field, the method to use is

```
txtDisplay.setText( btnOneText );
```

getText

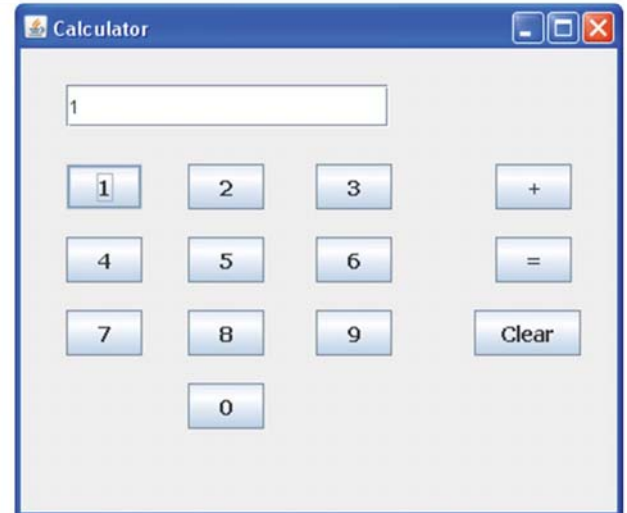
- To get text from a form object, you can use the **getText** method of the object (if it has one).
- So to get the text from **btnOne** we could use this code:

```
String btnOneText = btnOne.getText( );  
String textfieldText = txtDisplay.getText( );
```

- Add this code to your code stub:

```
private void btnOneActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String btnOneText = btnOne.getText();  
    txtDisplay.setText(btnOneText);  
}
```

- Run your program and test it out.
- Click your 1 button and you should find that the number 1 appears in the text field:



٤٣

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

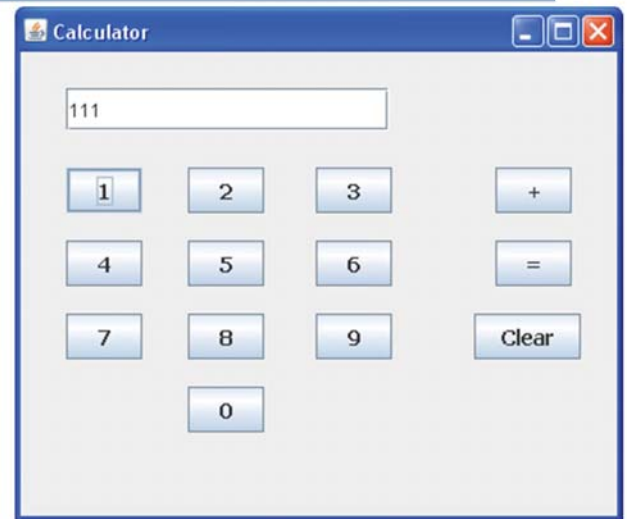
- There is a problem, however.
- What if you want to enter the number 11, or the number 111? When you click the button repeatedly, nothing else happens.
- It's always a 1, no matter how many times you click.
- To do that, you can simply get the text from the text field and combine it with the button text. Amend the first line of your code to this:

```
private void btnOneActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String btnOneText = txtDisplay.getText() + btnOne.getText();  
    txtDisplay.setText(btnOneText);  
}
```

٤٤

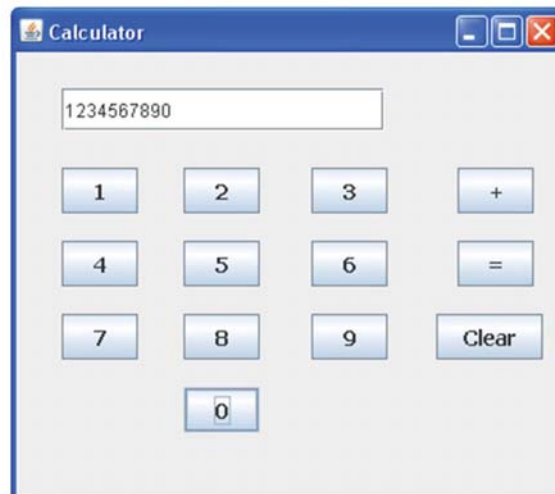
Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

- Repeat last step for the buttons from 0 to 9



```
private void btnOneActionPerformed(java.awt.event.ActionEvent evt) {  
  
    String btnOneText = txtDisplay.getText() + btnOne.getText();  
    txtDisplay.setText(btnOneText);  
}  
  
private void btnTwoActionPerformed(java.awt.event.ActionEvent evt) {  
  
    String btnTwoText = txtDisplay.getText() + btnTwo.getText();  
    txtDisplay.setText(btnTwoText);  
}  
  
private void btnThreeActionPerformed(java.awt.event.ActionEvent evt) {  
  
    String btnThreeText = txtDisplay.getText() + btnThree.getText();  
    txtDisplay.setText(btnThreeText);  
}  
  
private void btnFourActionPerformed(java.awt.event.ActionEvent evt) {  
    String btnFourText = txtDisplay.getText() + btnFour.getText();  
    txtDisplay.setText(btnFourText);  
}
```

- Run your calculator and test it out.
- You should be able to enter all the numbers from 0 to 9:



٤٧

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

```
private void btnFiveActionPerformed(java.awt.event.ActionEvent evt) {
    String btnFiveText = txtDisplay.getText() + btnFive.getText();
    txtDisplay.setText(btnFiveText);
}
```

```
private void btnSixActionPerformed(java.awt.event.ActionEvent evt) {
    String btnSixText = txtDisplay.getText() + btnSix.getText();
    txtDisplay.setText(btnSixText);
}
```

```
private void btnSevenActionPerformed(java.awt.event.ActionEvent evt) {
    String btnSevenText = txtDisplay.getText() + btnSeven.getText();
    txtDisplay.setText(btnSevenText);
}
```

```
private void btnEightActionPerformed(java.awt.event.ActionEvent evt) {
    String btnEightText = txtDisplay.getText() + btnEight.getText();
    txtDisplay.setText(btnEightText);
}
```

```
private void btnNineActionPerformed(java.awt.event.ActionEvent evt) {
    String btnNineText = txtDisplay.getText() + btnNine.getText();
    txtDisplay.setText(btnNineText);
}
```

```
private void btnZeroActionPerformed(java.awt.event.ActionEvent evt) {
    String btnZeroText = txtDisplay.getText() + btnZero.getText();
    txtDisplay.setText(btnZeroText);
}
```

٤٨

The plus bottom

- The next thing to do is to handle the plus button.
- The only thing our plus button needs to do is to store whatever number is currently in the text field.
- It's just a record of the first number to be added.
- To store the value, we need to set up a field variable, that's a variable outside of any button code.
- This is so that all the buttons can see what has been stored

```
public class JavaCalculator extends javax.swing.JFrame {  
  
    private double total1 = 0.0;  
  
    /** Creates new form JavaCalculator */  
    public JavaCalculator() {  
        ...  
    }  
  
    /** ... */  
}
```

49

-
- Double click your Plus button to generate the code stub.
 - Now add the following two lines to your plus button:

```
private void btnPlusActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    total1 = total1 + Double.parseDouble( txtDisplay.getText( ) );  
    txtDisplay.setText( "" );  
}
```

- Once the number is stored, we've cleared the text field.
- The user can now enter a second number to be added to the first.

The Equals Button

- After the user has chosen a second number, the equals button needs to be clicked.
- Clicking the equals button will produce the answer to the addition.
- To store the output of the calculation, we can set up another field variable.
- Add the following line to the top of your code:

```
public class JavaCalculator extends javax.swing.JFrame {  
  
    private double total1 = 0.0;  
    private double total2 = 0.0;  
  
    public JavaCalculator() {  
        initComponents();  
    }  
}
```

-
- To get the answer to the calculation, we take whatever is currently stored in total1 and add it to whatever is currently in the text field.
 - get the text from the text field and converts the string into a double.
 - The result is then added to total1.
 - The answer is then stored in the total2 variable.
 - display the answer to the calculation back into the text field.
 - clear the total1 variable

```
private void btnEqualsActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    total2 = total1 + Double.parseDouble( txtDisplay.getText( ) );  
    txtDisplay.setText( Double.toString( total2 ) );  
    total1 = 0;  
}
```

The Clear Button

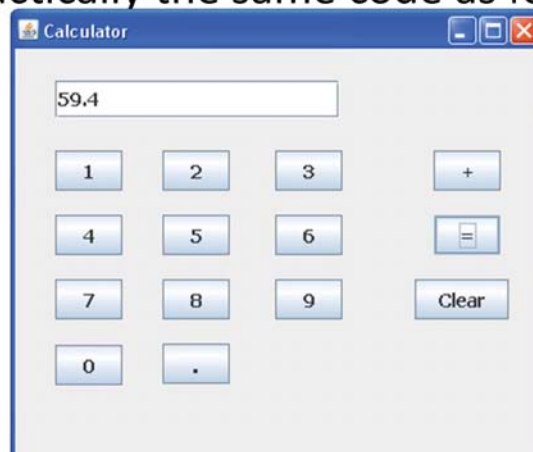
- The only thing left to code for now is the Clear button.
- For this, we need to clear the total2 variable and set the text in the text field to a blank string.

```
private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    total2 = 0;  
    txtDisplay.setText("");  
}
```

Exercises

Exercise 1

- There is one thing missing, however – a point symbol. At the moment, your calculator can't handle sums like $23.6 + 35.8$. So add a point button to your form.
- Write the code for it.
- (Hint – it's practically the same code as for your number buttons.)

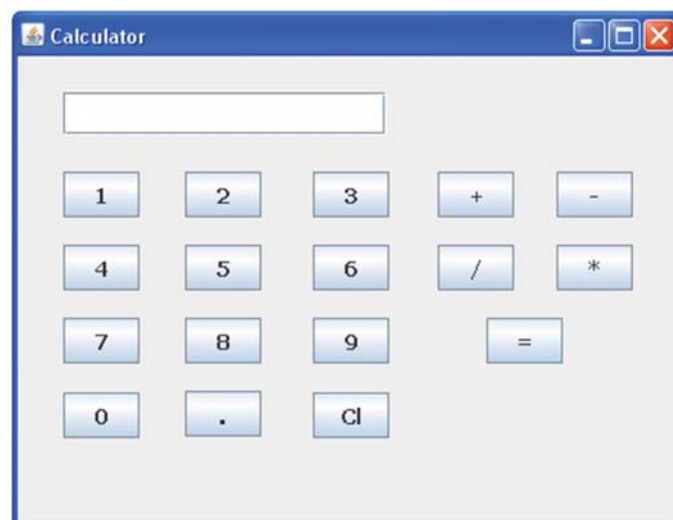


Extending the Functionality of your Calculator

- Now that the plus button is working OK, we can add buttons to subtract, divide and multiply.
- Just like the addition button, however, these buttons won't do any calculations: the equals button will still do all the work.
- The only thing the operator buttons will do is to record which button was clicked: add, subtract, divide, or multiply.
- The first thing to do is to place some more buttons on your form.
- In the image below, we've moved the Clear button, and put all the operator buttons in the panel to the right.
- Feel free to come up with your own design, though:

00

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013



01

Dr. Ahmed ElShafee, Fundamentals of Programming II, Fall
2013

-
- Once you've added the new buttons, rename the default variables to btnSubtract, btnDivide, and btnMultiply.
 - The technique we'll use to get which operator button was clicked is to store the button text in a field variable.
 - We can then use a switch statement to examine which character is in the field variable.
 - If it's the + symbol we can add; if it's the symbol we'll subtract; if it's the / symbol we'll divide; and if it's the * symbol we'll multiply.

-
- Add the following field variable to the top, just below your other two:

```
public class JavaCalculator extends javax.swing.JFrame {  
  
    private double total1 = 0.0;  
    private double total2 = 0.0;  
    private char math operator;  
  
    public JavaCalculator() {  
        initComponents();  
    }  
}
```

-
- We can set up a method to get at the character on the button that was clicked.
 - Add the following method to your code:

```
private void getOperator(String btnText) {  
    math_operator = btnText.charAt(0);  
    total1 = total1 + Double.parseDouble(txtDisplay.getText());  
    txtDisplay.setText("");  
}
```

- We've called the method getOperator.
- It's a void method, so won't return any value
- we have a String variable called btnText. This is obviously the text from the button that was clicked

-
- The charAt method of strings will get a character from a string.
 - The character you want goes between the round brackets of charAt.
 - The math symbol from our buttons is always at character 0 in the string.
 - This is then stored in the char field variable we've just set up at the top of the code.
 - Notice the other two lines in the code. They are exactly the same as the lines from the plus button, doing exactly the same thing – storing the first number in the variable called total1.

-
- So locate your **btnPlus** code and delete the following two lines from it:

```
private void btnPlusActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    total1 = total1 + Double.parseDouble( txtDisplay.getText( ) ) ;  
    txtDisplay.setText("");  
}
```

- Replace them with these two lines:

```
private void btnPlusActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String button_text = btnPlus.getText();  
    getOperator(button_text);  
}
```

-
- The same two lines can be added to the other operator button, only changing the name of the button.

```
private void btnSubtractActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String button_text = btnSubtract.getText();  
    getOperator(button_text);  
}
```

```
private void btnMultiplyActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String button_text = btnDivide.getText();  
    getOperator(button_text);  
}
```

```
private void btnMultiplyActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String button_text = btnDivide.getText();  
    getOperator(button_text);  
}
```

- For the equals button, we can set up a switch statement to examine what is in the **math_operator** variable.

```
switch ( math_operator ) {  
    case '+':  
        total2 = total1 + Double.parseDouble(txtDisplay.getText());  
        break;  
    case '-':  
        total2 = total1 - Double.parseDouble(txtDisplay.getText());  
        break;  
    case '/':  
        total2 = total1 / Double.parseDouble(txtDisplay.getText());  
        break;  
    case '*':  
        total2 = total1 * Double.parseDouble(txtDisplay.getText());  
        break;  
}
```

```
73 txtDisplay.setText(Double.toString(total2));  
    total1 = 0;
```

Enhancing Your calculator

```
public class From01 extends javax.swing.JFrame {  
    private double total1=0,total2=0;  
    private char op;  
    private Boolean newOp=true;
```

```

private void btnOneActionPerformed(java.awt.event.ActionEvent evt) {
    if (newOp)
    {
        txtDisplay.setText(btnOne.getText());
        newOp=false;
    }
    else
        txtDisplay.setText(txtDisplay.getText()+btnOne.getText());
}

private void btnTwoActionPerformed(java.awt.event.ActionEvent evt) {
    if (newOp)
    {
        txtDisplay.setText(btnTwo.getText());
        newOp=false;
    }
    else
        txtDisplay.setText(txtDisplay.getText()+btnTwo.getText());
}

```

```

private void btnEqualActionPerformed(java.awt.event.ActionEvent evt) {
    switch (op)
    {
        case '+':
            total2=total1+Double.parseDouble(txtDisplay.getText());
            break;
        case '-':
            total2=total1-Double.parseDouble(txtDisplay.getText());
            break;
        case 'X':
            total2=total1*Double.parseDouble(txtDisplay.getText());
            break;
        case '/':
            total2=total1/Double.parseDouble(txtDisplay.getText());
            break;
    }
    txtDisplay.setText(Double.toString(total2));
    total1=0;
    newOp=true;
}

```

assignment

- GPA Calculator

GPA Calculator for 1st Level

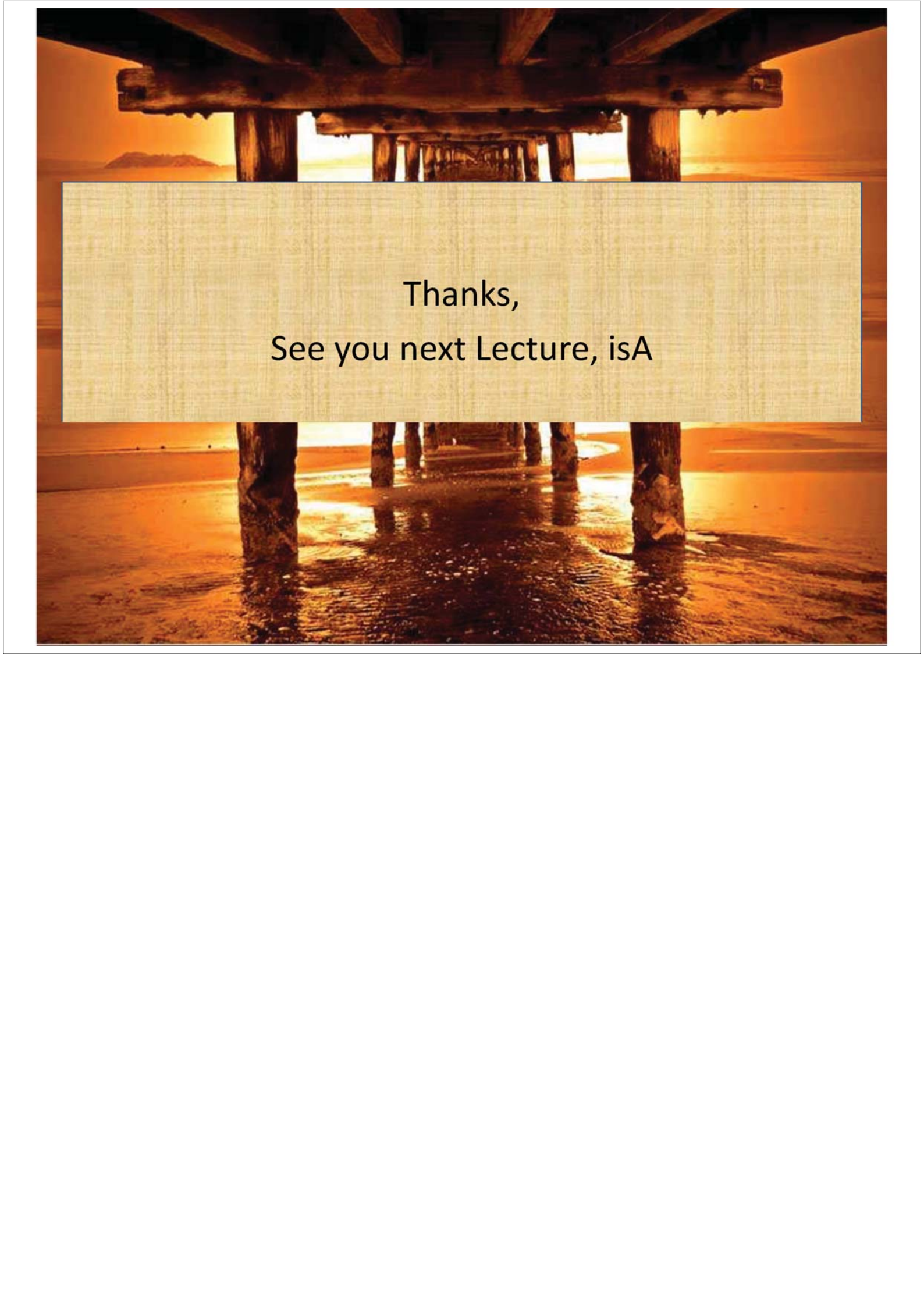
English 101	- ▼	English 102	- ▼
Algebra I	- ▼	Discrete	- ▼
Calculus I	- ▼	Calculus II	- ▼
Physics	- ▼	Electronics	- ▼
Managment	- ▼	Pre. Skills	- ▼
Intro. CS	- ▼	Fund. Prog. I	- ▼
		Critical Thinking	- ▼

Calculate

CH 0

GPA 0

Grade	GPA
A	4
A-	3.7
B+	3.3
B	3
B-	2.7
C+	2.3
C	2
C-	1.7
D+	1.3
D	1
F	0

A photograph of a wooden pier structure over water at sunset. The pier's wooden beams and pillars are visible, creating a perspective that leads towards the horizon. The water is calm, reflecting the warm orange and yellow light of the setting sun. A semi-transparent, light-colored rectangular box is centered over the image, containing the text "Thanks, See you next Lecture, isA".

Thanks,
See you next Lecture, isA