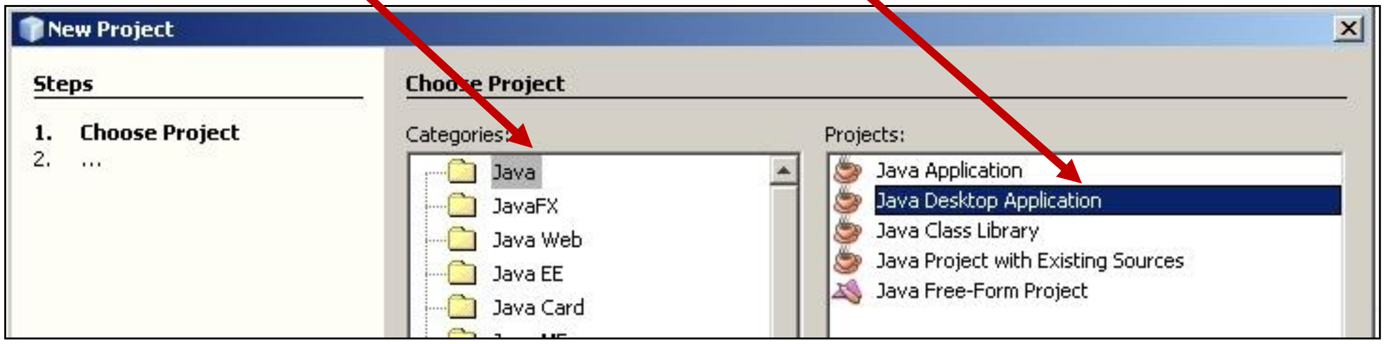
Advanced User Interfaces and Data Validation

1. Open **NetBeans** and click **File... New Project...**
   Select **Java** as the Category and **Java Desktop Application** as the Project
   Click **Next**.



New Project

Steps

1. Choose Project
2. ...

Choose Project

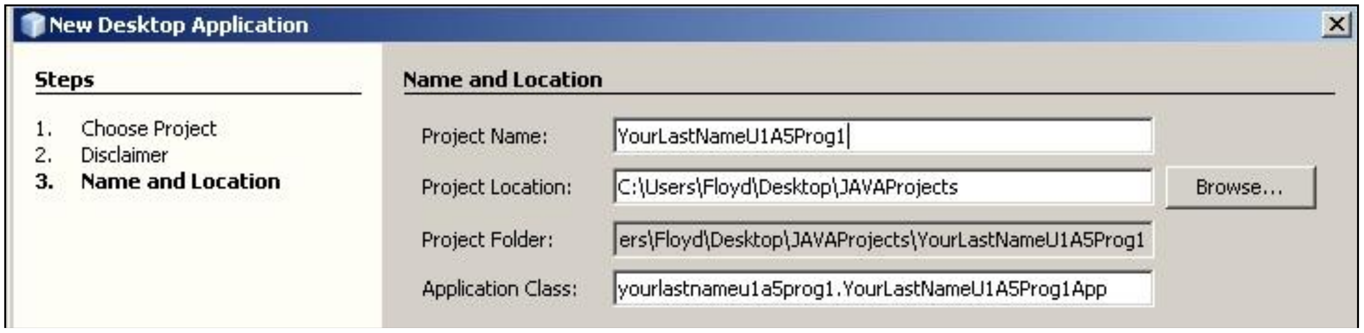
Categories:

- Java
- JavaFX
- Java Web
- Java EE
- Java Card

Projects:

- Java Application
- Java Desktop Application
- Java Class Library
- Java Project with Existing Sources
- Java Free-Form Project

2. Enter **YourLastNameU1A5Prog1** as the project name and click **Finish**.



New Desktop Application

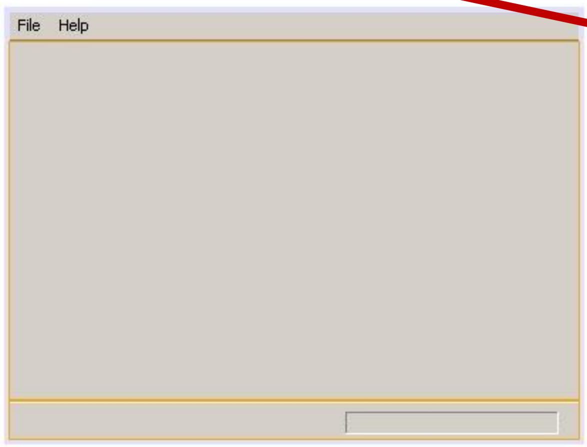
Steps

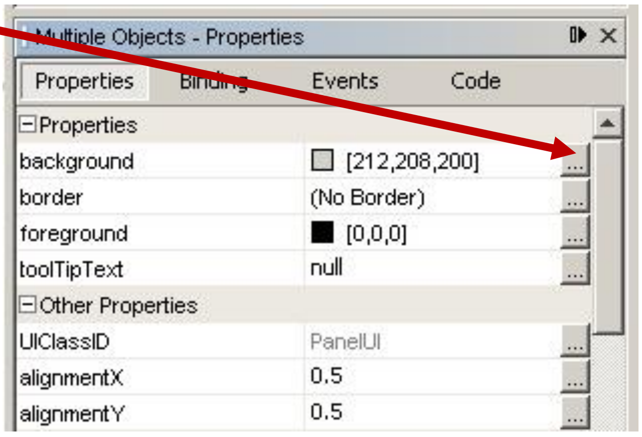
1. Choose Project
2. Disclaimer
3. Name and Location

Name and Location

Project Name:	YourLastNameU1A5Prog1
Project Location:	C:\Users\Floyd\Desktop\JAVAProjects
Project Folder:	ers\Floyd\Desktop\JAVAProjects\YourLastNameU1A5Prog1
Application Class:	yourlastnameu1a5prog1.YourLastNameU1A5Prog1App

3. The first thing we’re going to do is change the overall look of our user interface. With the main design area selected (yellow border around it), click on the three dots (ellipsis) next to **Background** in the Properties window:

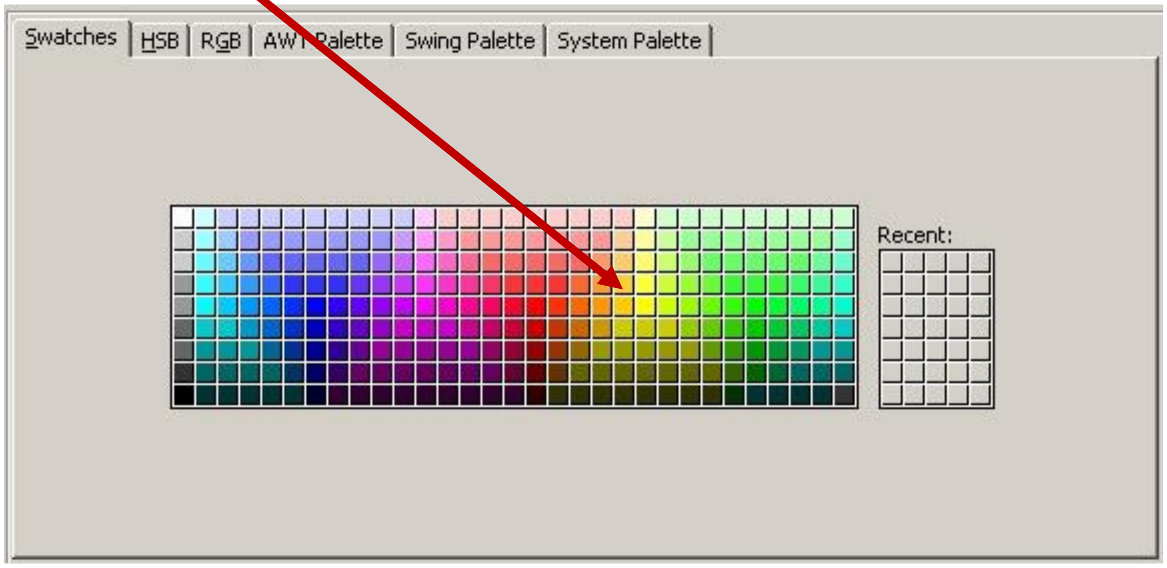



Multiple Objects - Properties

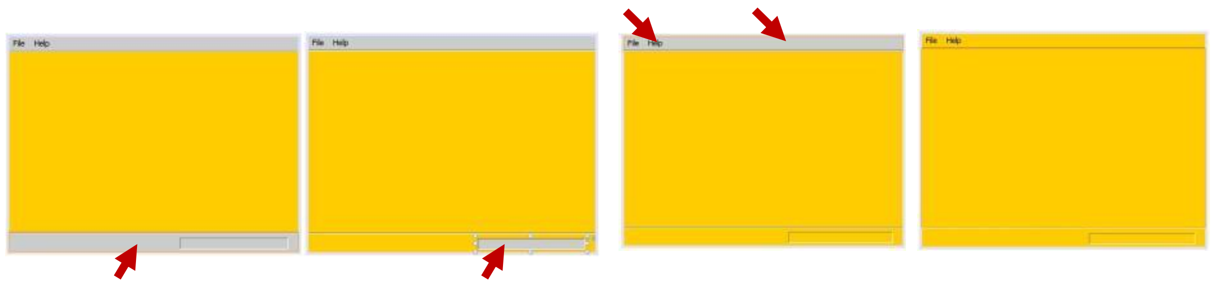
Properties | Binding | Events | Code

⊟Properties		
background	▢ [212,208,200]	...
border	(No Border)	...
foreground	■ [0,0,0]	...
toolTipText	null	...
⊟Other Properties		
UIClassID	PanelUI	...
alignmentX	0.5	...
alignmentY	0.5	...

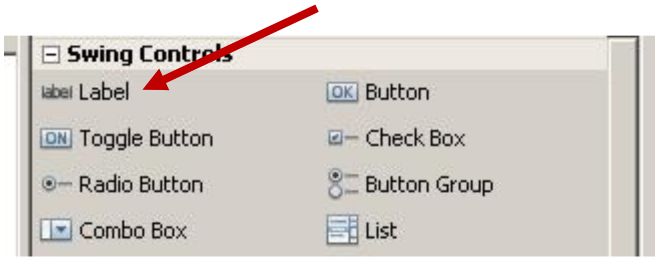
You should now be able to select a colour for your background. For now, select a light yellow and click **Okay**. The background of the main design area should now be yellow.



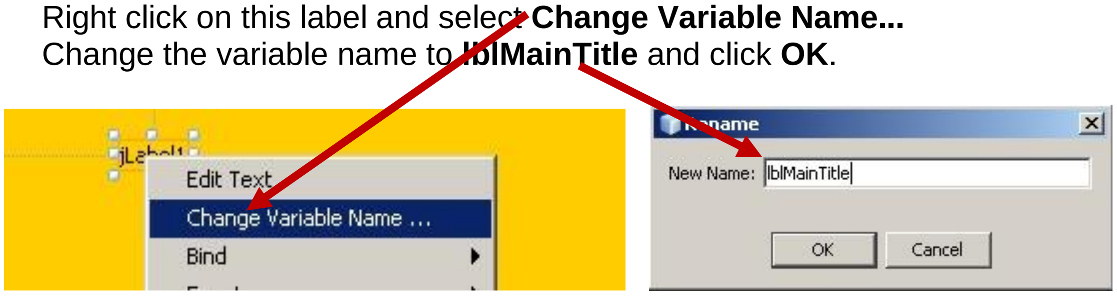
Repeat this process for the other parts of the program window. Click on each area and then alter the background colour until all visible parts have been changed to yellow.



4. We will now add a main title to our program.
   From the Swing controls click and drag a label onto the design area.

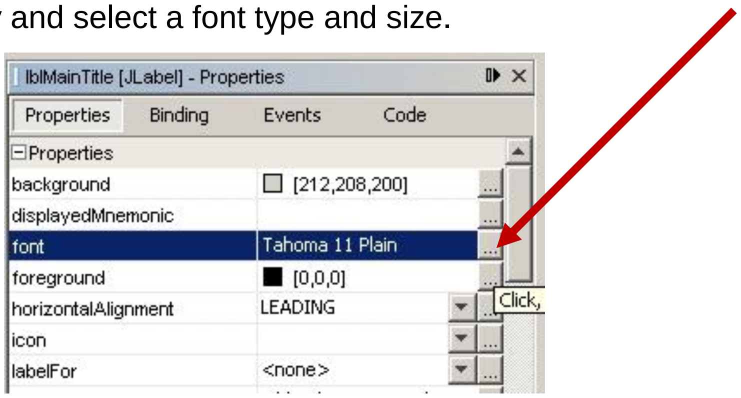


Right click on this label and select **Change Variable Name...**
Change the variable name to **lblMainTitle** and click **OK**.



Right click on this label again and select **Change Text.**
Type in **Bible Chapter Search**

To change the appearance of the main title of our program, select the ellipsis from next to the font property and select a font type and size.

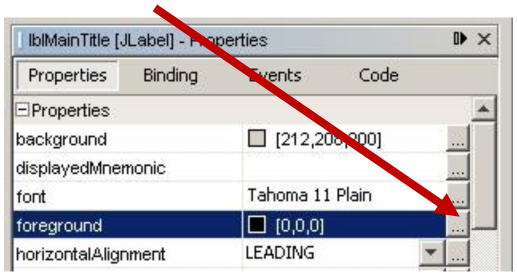


lblMainTitle [JLabel] - Properties

Properties | Binding | Events | Code

Properties	
background	[212,208,200]
displayedMnemonic	
font	Tahoma 11 Plain
foreground	■ [0,0,0]
horizontalAlignment	LEADING
icon	
labelFor	<none>

To alter the colour of the font colour, select the ellipsis next to the foreground property.

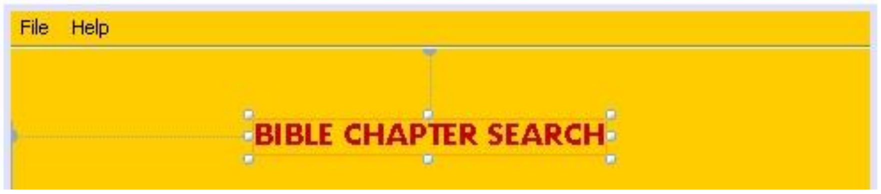


lblMainTitle [JLabel] - Properties

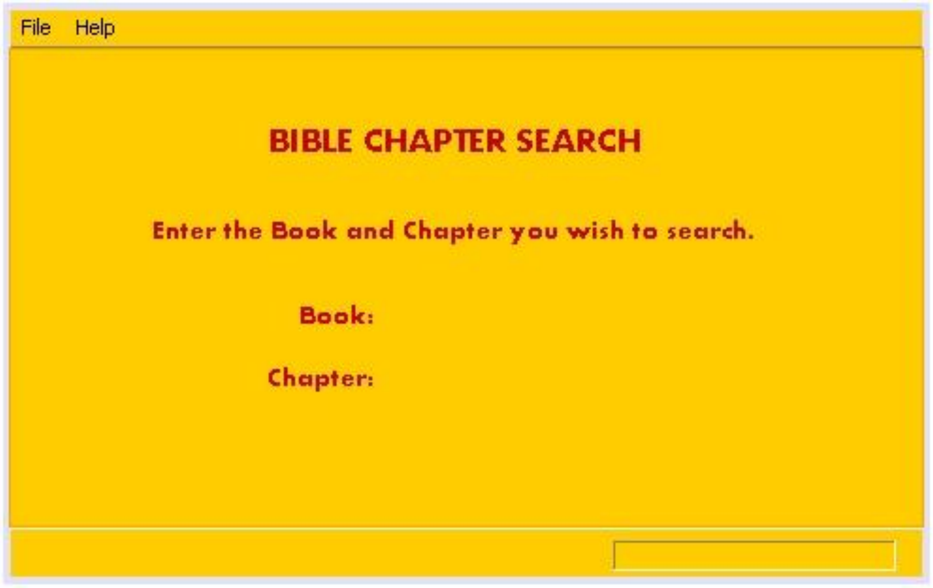
Properties | Binding | Events | Code

Properties	
background	[212,208,200]
displayedMnemonic	
font	Tahoma 11 Plain
foreground	■ [0,0,0]
horizontalAlignment	LEADING

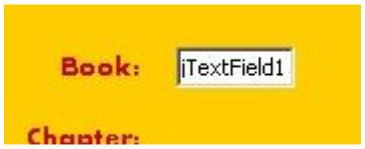
Alter your main title until it resembles the following (this is Aharoni, Plain, 18 pt font with a red foreground colour):



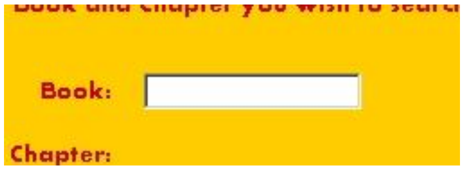
5. Add three other labels, so that your design window resembles the following (make sure you give your new labels appropriate names ie: lblInstruction, lblBook, lblChapter):



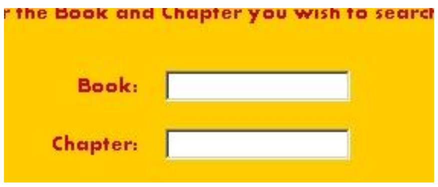
6. Add a text field to hold the user's input for the Book name. Drag a textbox from the Swing Controls onto the design area.



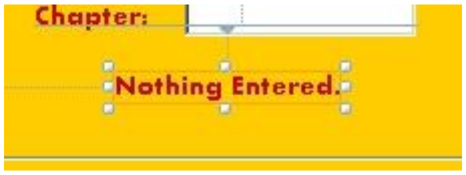
7. Right click on the text field and change the variable name to **txtBook**. Right click on the text field and click Edit Text. Delete the text so that nothing appears (after performing this step, the text field will probably become very small. You will have to drag it back to an appropriate size). Your text field should now resemble the following:



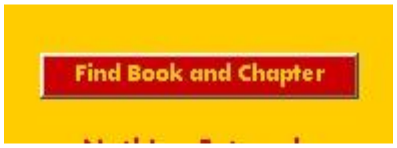
8. Add another text field to hold the Chapter. The variable name for this text field should be set to **txtChapter**. Once complete, it should resemble the following:



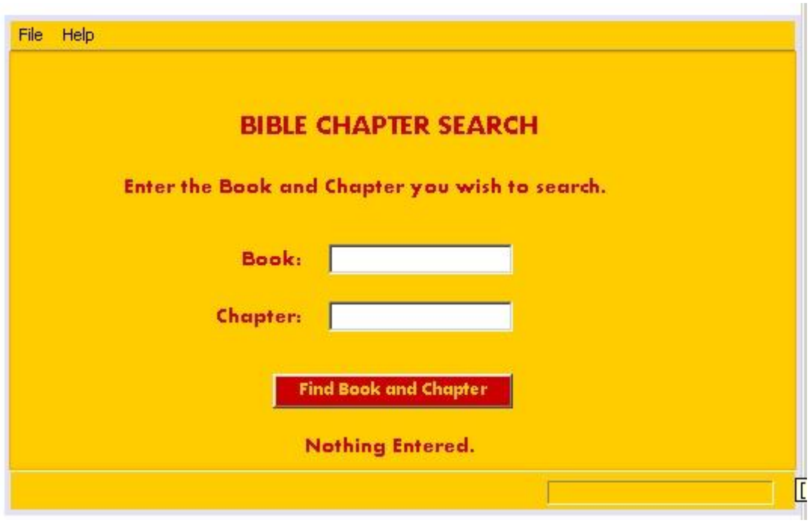
9. Add a final label to your design area. This label will hold the program output, once the user has entered a Book and Chapter. The variable name for the label should be set to **lblOutput** and it should contain the text "**Nothing Entered**". It should resemble the following:



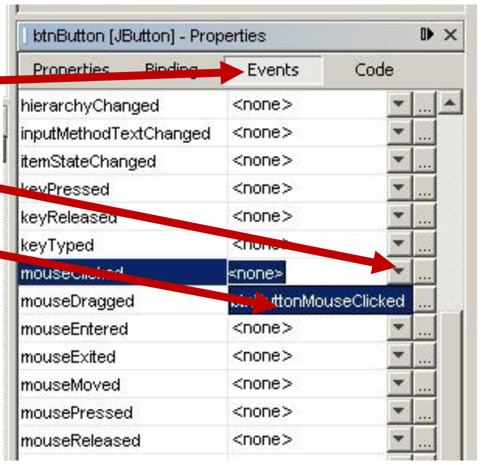
10. Finally, we need to enter the button that the user will click once they have entered a Book and Chapter number. Drag a button onto the design area and change the variable name to **btnButton** and the text to "**Find Book and Chapter**". Alter the foreground and font of the button so that it resembles the following:



11. Rearrange the components on your design area so that it resembles the following:

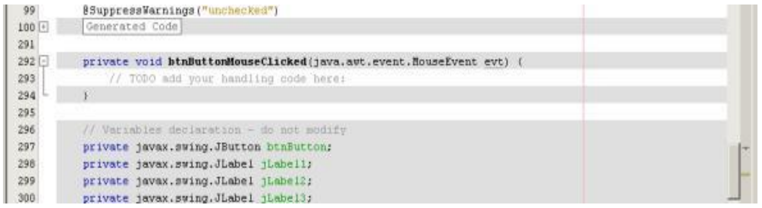


12. We are now going to add the code necessary to generate output when the button is clicked. Click on the **button** and then select **Events**.
Next to the **mouseClicked** event click on the **down** arrow.
Click on **btnButtonMouseClicked**.
This will take you to the area where you will enter the Java programming code.



btnButton [JButton] - Properties

hierarchyChanged	<none>
inputMethodTextChanged	<none>
itemStateChanged	<none>
keyPressed	<none>
keyReleased	<none>
keyTyped	<none>
mouseClicked	<none>
mouseDragged	btnButtonMouseClicked
mouseEntered	<none>
mouseExited	<none>
mouseMoved	<none>
mousePressed	<none>
mouseReleased	<none>



```
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnButtonMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnButton;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
```

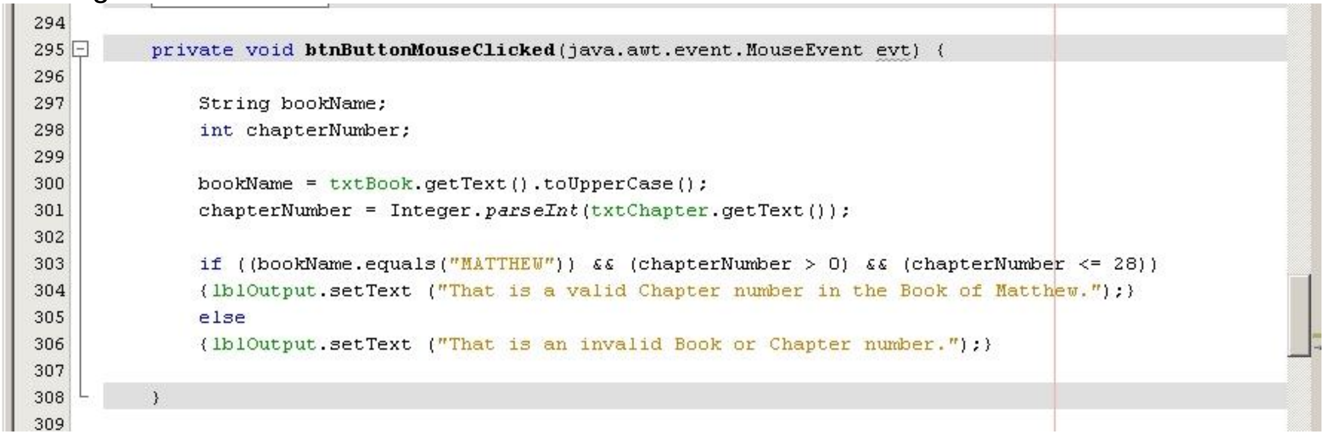
13. In the code window, we will enter the code that will generate output depending on the user's input.
The first few lines of code to be entered will ensure that the user's input is read into the program correctly, and altered so that it can be evaluated.
The first few lines of code you need to include are:

```
String bookName;
int chapterNumber;

bookName = txtBook.getText().toUpperCase();
chapterNumber = Integer.parseInt(txtChapter.getText());
```

The other code that needs to be added is the code that outputs a message to the user depending on the Book and Chapter entered.
The code below ensures that a valid Chapter number was entered for the book of Matthew:

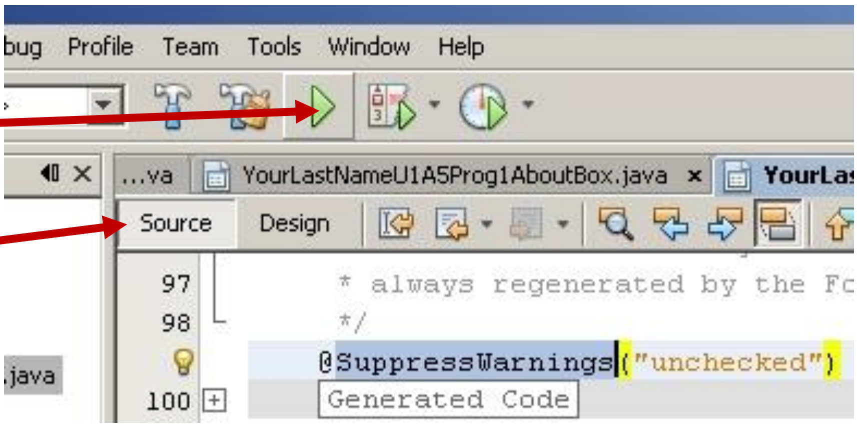
```
if ((bookName.equals("MATTHEW")) && (chapterNumber > 0) && (chapterNumber <= 28))
      {lblOutput.setText ("That is a valid Chapter number in the Book of Matthew.");}
else
      {lblOutput.setText ("That is an invalid Book or Chapter number.");}
```

You should enter the above code into the code window, so that is resembles the following:



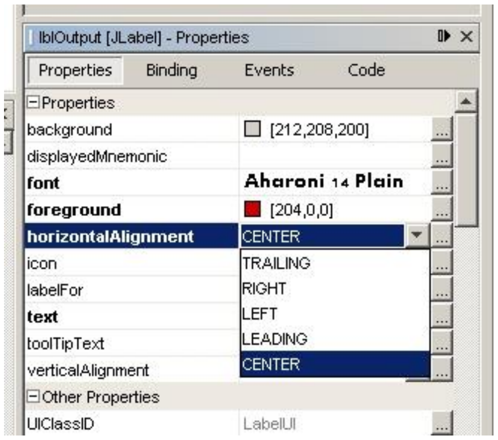
```
    private void btnButtonMouseClicked(java.awt.event.MouseEvent evt) {

        String bookName;
        int chapterNumber;

        bookName = txtBook.getText().toUpperCase();
        chapterNumber = Integer.parseInt(txtChapter.getText());

        if ((bookName.equals("MATTHEW")) && (chapterNumber > 0) && (chapterNumber <= 28))
        {lblOutput.setText ("That is a valid Chapter number in the Book of Matthew.");}
        else
        {lblOutput.setText ("That is an invalid Book or Chapter number.");}

    }
```

14. Once the code has been entered, you should be able to run your program by clicking on the **Run Main Program** button.
Note that if you want to jump back and forth between the **Source** and the **Design**, you just have to click the quick buttons at the top.
If your program doesn't run successfully then you may have to go back and check your code.


bug Profile Team Tools Window Help
...va
YourLastNameU1A5Prog1AboutBox.java
YourLas
Source
Design
.java

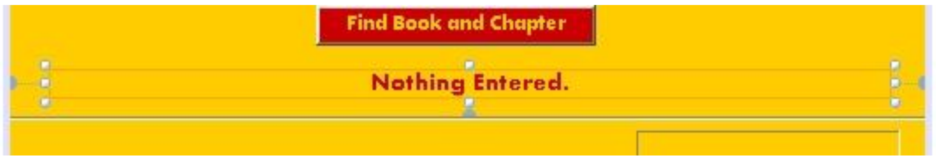
```
97      * always regenerated by the Fo
98      */
        @SuppressWarnings("unchecked")
100     Generated Code
```

15. As you run your program, you may have noticed that the output runs off of the program window and can't be read. We need to ensure that this doesn't happen. Go back to the design view of your program and click on the output label.
Stretch the size of the label so that it extends the entire width of the window, and then under the **properties** select **CENTER** for the **horizontalAlignment**.

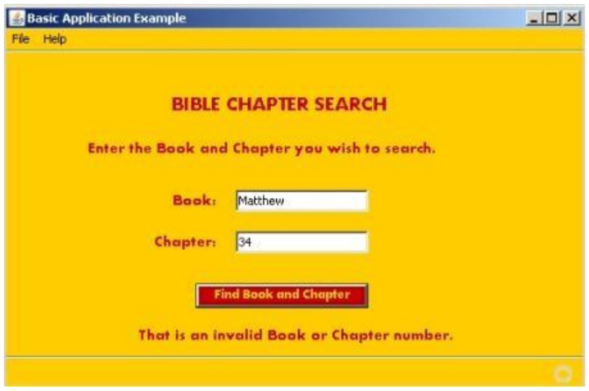


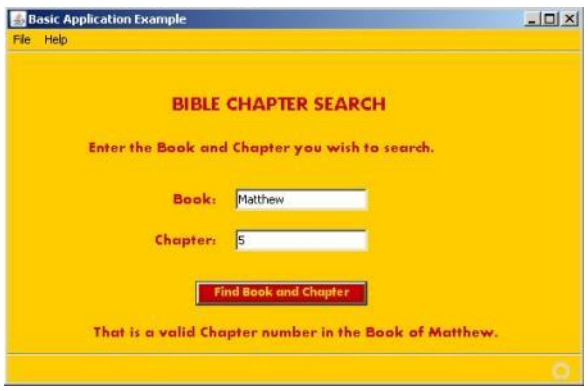
Properties	
background	[212,208,200]
displayedMnemonic	
font	Aharoni 14 Plain
foreground	🟥 [204,0,0]
horizontalAlignment	CENTER
icon	
labelFor	
text	
toolTipText	
verticalAlignment	
Other Properties	
UIClassID	LabelUI

Your output label should now resemble the following:



16. When you run your program, the output generated should resemble the following, depending on the input:


Basic Application Example
File Help
BIBLE CHAPTER SEARCH
Enter the Book and Chapter you wish to search.
Book: Matthew
Chapter: 34
Find Book and Chapter
That is an invalid Book or Chapter number.


Basic Application Example
File Help
BIBLE CHAPTER SEARCH
Enter the Book and Chapter you wish to search.
Book: Matthew
Chapter: 5
Find Book and Chapter
That is a valid Chapter number in the Book of Matthew.

17. Now that your program can effectively test input for the Book of Matthew, it is your task to add the code necessary to test other input.

    If the user enters “MARK” and a chapter that is between 1 and 16, then the output generated will read: “That is a valid Chapter number in the Book of Mark.”

    If the user enters “LUKE” and a chapter that is between 1 and 24, then the output generated will read: “That is a valid Chapter number in the Book of Luke.”

    If the user enters “JOHN” and a chapter that is between 1 and 21, then the output generated will read: “That is a valid Chapter number in the Book of John.”

    The else clause you created should be able to cover all other possible input, including a Chapter that does not fall within a Books range, as well as a Book name that isn’t Matthew, Mark, Luke or John.

Your program doesn’t actually have to retrieve the chapter, this assignment is meant to help you practice you input validation skills.