

Section 1: Define

Applet-

A **Java applet** is a [small application](#) which is written in Java and delivered to users in the form of [bytecode](#). The user launches the Java applet from a [web page](#), and the applet is then executed within a [Java Virtual Machine](#) (JVM) in a [process](#) separate from the [web browser](#) itself. A Java applet can appear in a frame of the web page, a new application window, [Sun's AppletViewer](#), or a stand-alone tool for testing applets.

Why does the an applet not have a main method?

It is actually good design but not obvious and what you want to do would have no effect so it is a little counter intuitive.

These types of applications live their lives in containers and as such their entry points are determined by the standards those containers must adhere to. The designers of these standards chose not to call the entry point main. You would place your functionality in an overridden method. All applets have the following four methods:

```
public void init();  
public void start();  
public void stop();  
public void destroy();
```

They have these methods because their superclass, `java.applet.Applet`, has these methods.

The superclass does not have anything but dummy code in these:

```
public void init() {}
```

If you want to derive a class to extend or change the name of `init()` you should implement your class and have your method call `init()`. This would use polymorphism to let you call the method whatever you like. Unless you are writing servlet container you are likely wasting your time.

Applets and Servlets do not start their own process. Instead they run inside a container. Therefore, they do not need a static main method (which starts the process), but a way to interact with their container.

Yes, but applets aren't applications. There *is* a main method in the applet runner (assuming it's implemented in Java; it need not be) but the applet doesn't work that way; it gets loaded/instantiated from a file and then it proceeds along its lifecycle through [initialization](#), [starting](#), operating, [stopping](#), and finally being [destroyed](#). The code that sends it through these states is hidden from the applet's view; it just knows it's in an [environment](#) that can run applets.

init-

This method is intended for whatever initialization is needed for your applet. It is called after the param tags inside the applet tag have been processed.

start-

This method is automatically called after the browser calls the init method. It is also called whenever the user returns to the page containing the applet after having gone off to other pages.

stop-

This method is automatically called when the user moves off the page on which the applet sits. It can, therefore, be called repeatedly in the same applet.

destroy-

This method is only called when the browser shuts down normally. Because applets are meant to live on an HTML page, you should not normally leave resources behind after a user leaves the page that contains the applet.

Task 1:

USE OBJECT ORIENTATED PROGRAM DESIGN TO SOLVE PROBLEM

Create basic Java Applet, based upon assignment #4, Task 2.

Launch your applet in a web browser.

Display your Applet in a very basic HTML webpage.

Attach Snipping photos below.

```

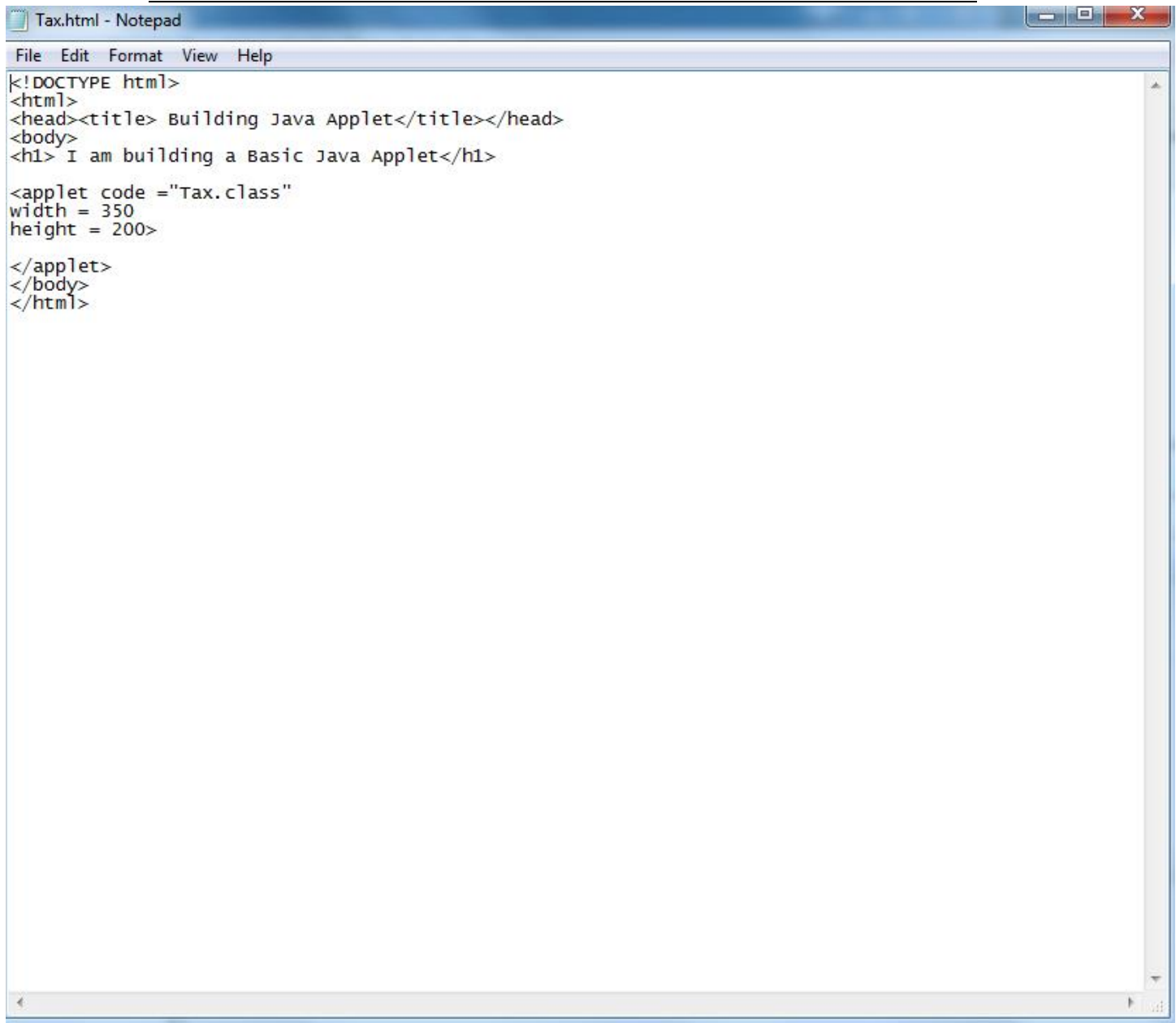
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6
7  package javaapplication1;
8
9
10 import java.awt.*;
11 import java.awt.ActiveEvent;
12 import java.awt.event.ActionEvent;
13 import java.awt.event.ActionListener;
14 import javax.swing.JApplet;
15 import javax.swing.*;
16 import javax.swing.border.TitledBorder;
17
18 public class Tax extends JApplet implements ActionListener{
19
20     private JTextField taxRate = new JTextField();
21     private JTextField computeTax = new JTextField();
22
23     private JButton compute;
24
25     public void init(){
26         JPanel p1 = new JPanel(new GridLayout(5,5));
27
28         taxRate.setHorizontalAlignment(JTextField.RIGHT);
29         computeTax.setHorizontalAlignment(JTextField.RIGHT);
30         computeTax.setEditable(false);
31
32         p1.setBorder(new TitledBorder("Please Enter your tax"));
33         p1.add(new JLabel("Your TaxRate"));
34
35         p1.add(taxRate);
36         p1.add(new JLabel("Tax Result"));
37         p1.add(computeTax);
38
39         JPanel p2 = new JPanel(new FlowLayout(FlowLayout.RIGHT));
40         compute = new JButton("Compute Tax");

```

```

41     compute.addActionListener(this);
42     p2.add(compute);
43
44     add(p1, BorderLayout.CENTER);
45     add(p2, BorderLayout.SOUTH);
46 }
47
48 public void actionPerformed(ActionEvent ae) {
49     double tax = Double.parseDouble(taxRate.getText());
50
51     if(tax < 0) {
52         computeTax.setText("Invalid input");
53     } else if (tax >= 0) {
54         if(tax < 6001) {
55             tax = tax * 0.10;
56         } else if (tax >= 6001 && tax <= 27950) {
57             tax = tax * 0.15;
58         } else if (tax >= 27951 && tax <= 67700) {
59             tax = tax * 0.27;
60         } else if (tax >= 67701 && tax <= 141250) {
61             tax = tax * 0.30;
62         } else if (tax >= 141251 && tax <= 307050) {
63             tax = tax * 0.35;
64         } else if (tax > 307051) {
65             tax = tax * 0.386;
66         }
67     }
68
69     computeTax.setText(Double.toString(tax));
70
71 }
72
73 }
74

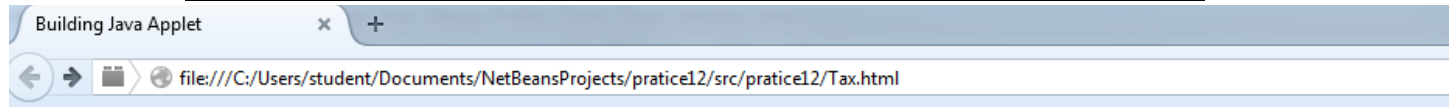
```



```
File Edit Format View Help
<!DOCTYPE html>
<html>
<head><title> Building Java Applet</title></head>
<body>
<h1> I am building a Basic Java Applet</h1>

<applet code ="Tax.class"
width = 350
height = 200>

</applet>
</body>
</html>
```



I am building a Basic Java Applet

