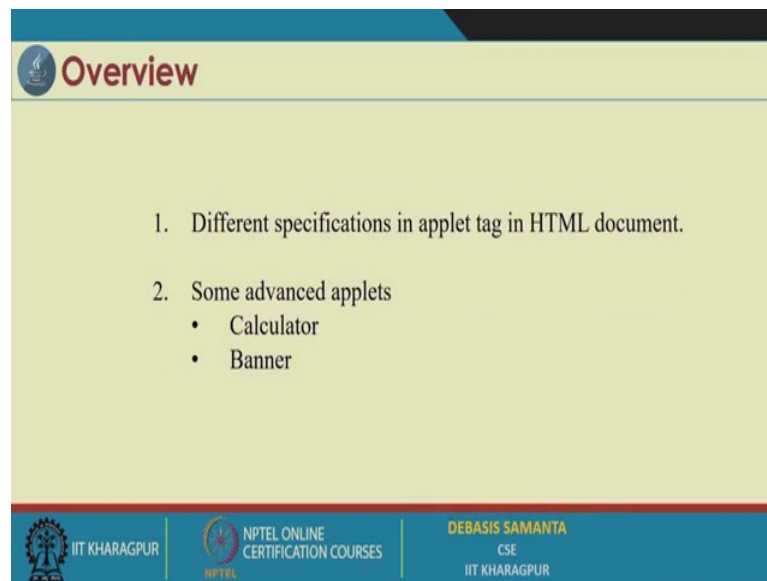


Programming in Java
Prof. Debasis Samanta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 38
Demonstration – XIV

Now, here is the second demonstration on the applet programming.

(Refer Slide Time: 00:22)



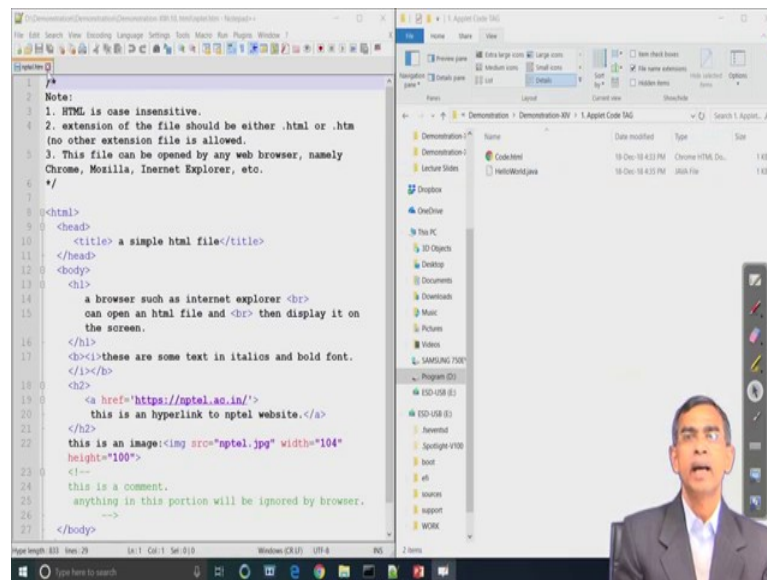
Overview

1. Different specifications in applet tag in HTML document.
2. Some advanced applets
 - Calculator
 - Banner

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASIS SAMANTA CSE IIT KHARAGPUR

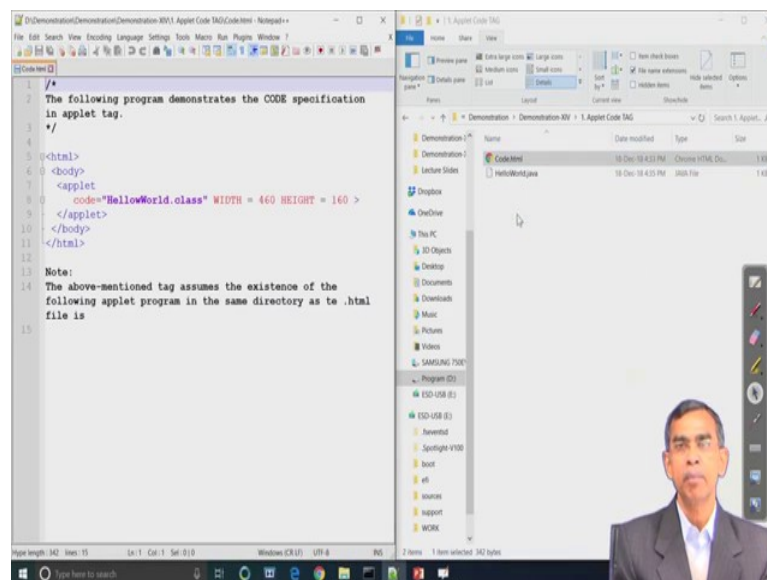
In these demonstrations we are going to highlights about the specification of HTML documents and then, we will conclude this demonstration with two applets; one is the calculator and the banner applets. Some let us have the HTML contents is there.

(Refer Slide Time: 00:40)



Now as you know HTML is coming into the purview of applet programming because the applet code should be hosted by an HTML page.

(Refer Slide Time: 00:50)

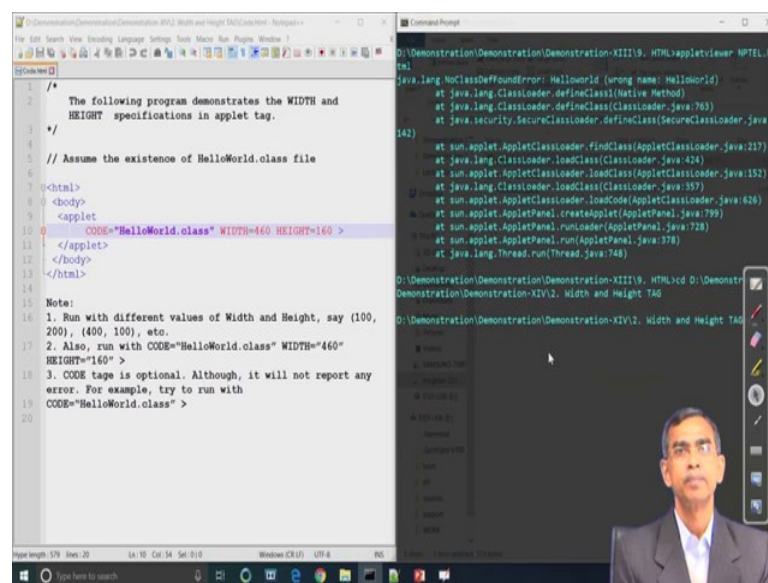


And in an HTML page the applet tag is there as you see here in this particular example here this is basically the HTML page content and then, applet tag it is there. Now, this the applet tag is coming here in its simplest form, but an applet tag is there with many other parameter specifications. Now, we will discuss one by one those are the parameters which are more relevant. So, for the applet programming, it is concerned. Now, the first

parameter that is there is the code itself and then, width and height these are the masked parameter. The mandatory parameter any an applet tag these two parameters I mean code and width and height are to be there. If there is an applet tag without this, this means that it is an invalid or incorrect applet tag.

Now, in addition to this valid and then necessary parameters, there are few more parameters are there and the width and height as you have already shown and you know exactly what is the meaning of width and height.

(Refer Slide Time: 02:05)



The screenshot shows a web browser window on the left and a command prompt window on the right. The browser window displays an HTML file with the following code:

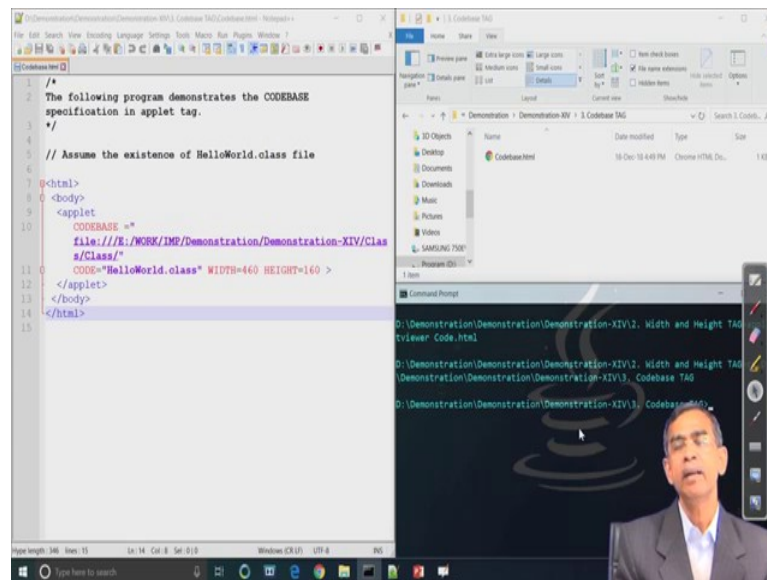
```
1  /*
2  3  The following program demonstrates the WIDTH and
4  5  HEIGHT specifications in applet tag.
6  7  */
8  9  // Assume the existence of HelloWorld.class file
10 11 <html>
12 13 <body>
14 15 <applet
16 17     CODE="HelloWorld.class" WIDTH=460 HEIGHT=160 >
18 19 </applet>
19 20 </body>
20 21 </html>
```

The command prompt window shows the execution of the applet, displaying a stack trace for a `NoClassDefFoundError`:

```
D:\Demonstration\Demonstration\Demonstration-XIII\9. HTML\appletviewer NPTEL.
.html
java.lang.NoClassDefFoundError: HelloWorld (wrong name: HelloWorld)
    at java.lang.ClassLoader.defineClass(Native Method)
    at java.lang.ClassLoader.defineClass(ClassLoader.java:763)
    at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:
142)
    at sun.applet.AppletClassLoader.findClass(AppletClassLoader.java:217)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:424)
    at sun.applet.AppletClassLoader.loadClass(AppletClassLoader.java:152)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:357)
    at sun.applet.AppletClassLoader.loadCode(AppletClassLoader.java:626)
    at sun.applet.AppletPanel.createApplet(AppletPanel.java:799)
    at sun.applet.AppletPanel.run(AppletPanel.java:728)
    at sun.applet.AppletPanel.run(AppletPanel.java:378)
    at java.lang.Thread.run(Thread.java:748)
```

This basically whenever an applet we want to display, it is fluidic in the sense that its size can be specified by these values. All these values are expressed in pixel. So, 460 by 160 pixel-like and we can control the size of the applet by showing the changing these values there. Now for example, here 460 and 160 being the applet size, it will give and display and then let us have the display with this 460 and 160 it is there,

(Refer Slide Time: 02:39)



So, this basically total if this is the total computer resolution is 1020 divided by 1020 whatever it is there, then it basically coming into that specification there. Now, let us change this again HTML page a little bit by say 200 by 200. As you see the side will change again. We have just do not change any code. Only you have changed the HTML file, save this HTML file, again the applet we are no need to compile it again as you see the applet yes come. So, an applet has changed its size. So, this is basically the parameter width and height by which the size of an applet 2 if it can be controlled.

Now, our next important tag that is possible next important parameters the specification that is possible in an applet is called the codebase. Now, codebase basically the idea it is that the class file that needs to be executed by an applet viewer or browser should be available somewhere here and there. So, codebase is basically said were this the location of your class file. In this case, we can see our HTML file and then, class file by default if they present in the same directory, then no code base specification is required.

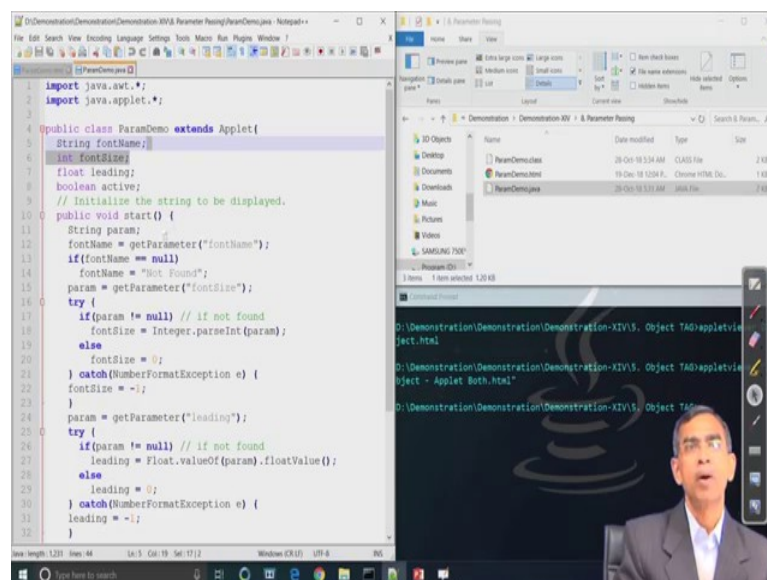
On the other hand, if your file is in one directory for example here in class directory whereas, your HTML file is in some other directory like in this case as you see in the different directory, then we have to give the relative location of your base and then formal finally the class declaration.

So, here again you see in the codebase the value that we have specified using this file is in E drive. One directory is there under this directory, another subdirectory, then there is

subdirectory class within this class this file is available. So, this is a complete specification about the location of the .class file which is there and it will be able to locate it and then, execute it.

In alternatively the here basically in the same machine if it is the same machine, but in different directories; on the other hand, if it is not in the same machine, the browser will browse from your own machine is a local right browser machine. And your this class file can be from a remote machine in any server or in a remote PC and then, we can and this machine it is connected to the net, then we can give the complete location of the URL of that machine then and then the class file. So, it basically browser will browser or applet viewer will probe into that remote machine.

(Refer Slide Time: 05:59)



And then we will access it, so, this is the idea about the codebase. Now, archive is another example ok. You can have this demo as you see we have changed the codebase. Here we can maintain the HTML file in different directories and then, class file in a different directory giving this specification will find its own source and then it will run it.

Now the archive is another right now in your code, in your applet tag or in your applet rather if it needs many other things to be imported from a distant location and every time it basically refers to an import item, then it needs to be connected to the machine and then, fetch it and then execute it.

This will claim time communication over. It also will increase in order to avoid it in the applet took. There is a specification called the archive. Archive basically whatever the files or items that is required to run this applet can be punched together in one jar file and then, the same file can be pasted or can be imported only once and find it.

Here, for example, our source file is .class file is here which is there and some other files it is there which is required there and this file is put in the jar file there. Now you will go to this there and then, jar file location you can see it there. Here is the jar file go to the jar file and you can see the jar file contains many other important necessary file it is there ok.

So, right we can find it and then it will basically as you have made it compressed also, so, compression is good. Now so, it will basically all components will be pasted once and then, automatically from the jar it will find its own file. If it does not file, it will not do the correct execution of course. So, this is the concept of archive it is there.

Then our next is object specification as you know the object is an alternative tag specification in an HTML 5 which basically supported by HTML 5 version and upward compatible language actually and it is basically the object is basically the same as the applet. Now, here you can see in live of applet the object tag is used there and if we run it applet viewer, as well as the browser all, can recognize this object now because we are using this browse right applet viewer.

So, this basically you see here no object code, object tag, and no applet tag, but object tag and within the object tag we can do it. So, it is basically HTML version compatibility and sometimes to make the version compatible irrespective of the different HTML versions that a browser can support. We can use both the code together.

Here is an example. So, object and applet code can be nested in the same HTML file, so that whatever there is a browser, it can support whether HTML version 4 or 5 whatever it is, it will basically support and this is an example as you see object code is there, a plate code is there which is basically applicable relevant to a browser it will get it and then do it.

So, this is the one HTML file is an alternative way that is basically compatible with any version of HTML page. As you see here it is basically executing it basically gives a

thread to whichever is applicable. So, an object is first encounter object is started and then, it will basically HelloWorld class code will be executed properly.

So, this is the object tag is there, there are some more tags. All tags all specification and the v space h space name and everything name are there. Name actually you can mention the name of an applet that can be used for an internal purpose of applet execution when they are in a distributed environment.

That means, 3-4 applets are running concurrently, but from the distance machines and they can communicate with each other and this process of communication protocol applet names needs to be recognized by which the applet should be specified by this applet tag. The name of each applet here actually in this applet tag we can mention name. Let the name of this writer code there right no here right name.

So, just simply give the name specification name this equals within double quote and you can give the name XYZ whatever it is there. So, these basically specify the name of the current applet that needs to be executed like this way and this is basically networking communication and everything which basically we are not in a position to discuss. So, I just want to skip this discussion.

And then, few other applet specifications which basically not so much relevant in the context of appletviewer usually do not consider all those things like v space h space align all these things. Those are coming in the purview of HTML browsing. If you have the internet explorer version 4 and onwards, then you can understand about it this basically positioning the applet and then in a proper or in an aesthetic manner.

So, that applet we will look looks good. So far is the for a feel-good impression applet can be managed or it can be arranged and for arranging to align v space and h space is there as it is HTML specific and also not specific to applet viewer. It is better to skip it there. Now let us come to the end of this one; I just want to conclude these things by giving details about more sophisticated and more useful programs first we have to have the idea about.

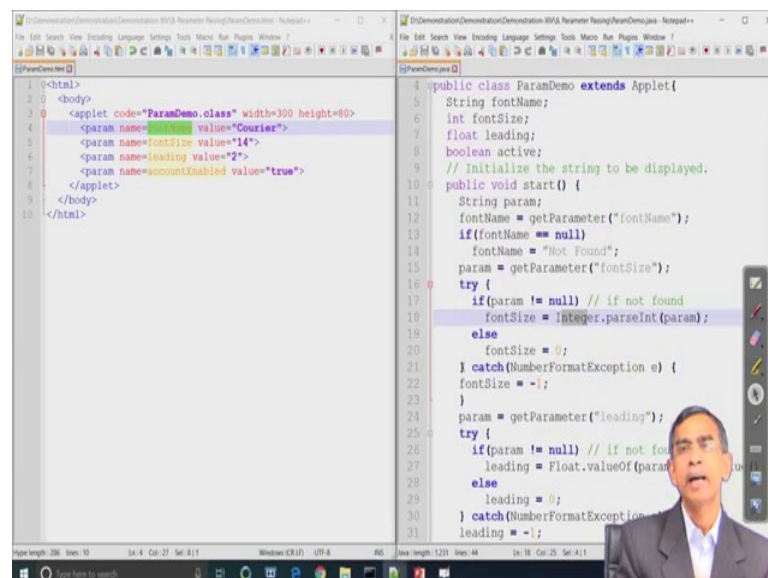
One thing I have just skipped it as a parameter passing and this parameter passing can be done by the param specification within the applet tag. Now, here I can see this is the one applet class, the ParamDemo .class and then this class is basically yes and then you see

this param class we will get the parameters here. The parameters are named as font name, font size and then, leading and account enabled and their values also supplied side by side like for this first param name.

The font name value is courier we can include in double quote whatever it is there. So, double quote please yeah. So, easily it is customized to include all the parameters in a double quote because HTML we will treat all these parameter values as a string and then, so all these things will be treated as a string and then and then basically than in the HTML they should be specified here.

Now, no it is not like that what you have done it is now let us pass the parameter ParamDemo java class ok. We can come to the HTML file here, that is ok. This is the HTML file. Now let us have the class file for this one. Here is a class file as you see font name font size leading and boolean actives are the 4 parameters which is basically local to this class we have declared here and those values you see how we can extract it either using init method or in the start method whatever it is there. Now in the start method, we have used this param basically the temporary temporarily string one and then you see it gets its values font name from the parameter.

(Refer Slide Time: 14:36)



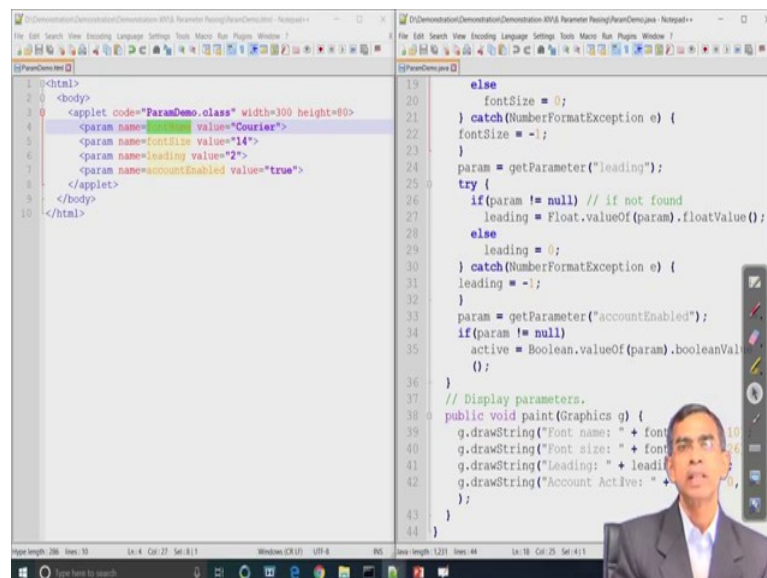
The screenshot displays two side-by-side code editors. The left editor shows an HTML file with an applet tag that includes four parameters: 'fontName' with value 'Courier', 'fontSize' with value '14', 'leading' with value '2', and 'accountEnabled' with value 'true'. The right editor shows the corresponding Java code for the 'ParamDemo' class, which extends 'Applet'. The code defines four instance variables: 'fontName', 'fontSize', 'leading', and 'active'. The 'start()' method uses 'getParameter()' to retrieve these values. For 'fontSize', it includes a try-catch block to handle 'NumberFormatException' if the value is not an integer. For 'leading', it uses 'Float.valueOf()' and also includes a try-catch for 'NumberFormatException'.

Now here in this HTML, this hosts the parameter .class which basically the compiled version of this one and the param name font name. This parameter font name is basically Courier and here you see font name get parameter, this method get parameter is defined

in applet class. This basically gets these values of this font name which basically here and then, Courier and you see the name which is there in the font name in the HTML file and within this double quote, these should match.

If there is no matching, then it will basically gives an error. To handle this error we usually put a try-catch block. Now so, this will be just font name. This parameter is get the value here the value. For example, Courier and it will store in the param parameter he as a string and from this string we have to process it either into the integer. For example, for this font size, the value should be passed as integer here, fontsize for others. For example, leading value also integer we have to pass into integer and for this action enable is a Boolean its value to be passed into Boolean value, right.

(Refer Slide Time: 15:55)



The image shows two Notepad++ windows side-by-side. The left window, titled 'ParamDemo.html', contains HTML code for an applet. The right window, titled 'ParamDemo.java', contains the corresponding Java code. A small video inset of a man is visible in the bottom right corner of the Java code window.

```
1 <html>
2 <body>
3 <applet code="ParamDemo.class" width=300 height=80>
4   <param name="fontName" value="Courier">
5   <param name="fontSize" value="14">
6   <param name="leading" value="2">
7   <param name="accountEnabled" value="true">
8 </applet>
9 </body>
10 </html>
```

```
19 else
20     fontSize = 0;
21 } catch (NumberFormatException e) {
22     fontSize = -1;
23 }
24 param = getParameter("leading");
25 try {
26     if (param != null) // if not found
27         leading = Float.valueOf(param).floatValue();
28     else
29         leading = 0;
30 } catch (NumberFormatException e) {
31     leading = -1;
32 }
33 param = getParameter("accountEnabled");
34 if (param != null)
35     active = Boolean.valueOf(param).booleanValue();
36 }
37 // Display parameters.
38 public void paint(Graphics g) {
39     g.drawString("Font name: " + fontName, 10, 10);
40     g.drawString("Font size: " + fontSize, 10, 20);
41     g.drawString("Leading: " + leading, 10, 30);
42     g.drawString("Account Active: " + active, 10, 40);
43 }
44 }
```

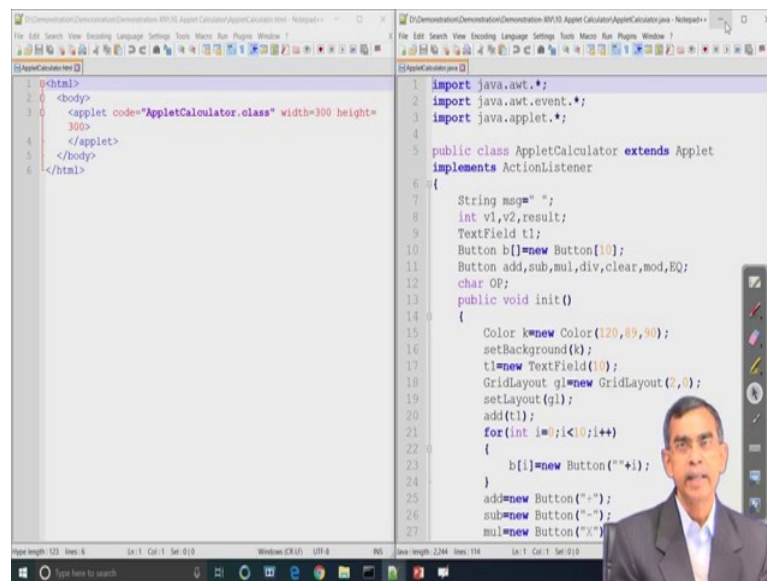
So, string to Boolean conversion and store inactive. So, what we can see is that the HTML we supply the different parameters which can be extracted by an applet while it in execution and then, gets it values and then this value can be used in the paint method or some other method to process it. Here we are using all those values to simply display it.

So, this way we can pass the parameters from HTML to an applet execution and this parameter passing is very much important because we cannot give direct input from our console keyboard or any other input to the applet as you see this applet got the value and then display it anyway. So, this is the idea of the parameter passing techniques. We have

an idea about the applet construction by virtue of many methods that are there overwriting those methods which are defined in abstract class.

Now, let us come to the demonstration of the calculator applet. So, a calculator applet typically we will look looks like this here let us have the first a look of a calculator that we are going to design it here, run this calculator program.

(Refer Slide Time: 17:15)



So, that we can have the look of this, then we can explain it here. Yes now, this is a very simple and innocent look of a calculator because I just want to make it more simple so that we can understand about it in this calculator. What we see there are 10 buttons numbers digits 0 to 9. So, basically, these 10 buttons are essential to input 10 numbers I mean digits I mean typing from the keyboard.

So, if we type two it will go into this one and then, like this one. As you see here whenever we click a particular button actually you can say button two, right. It will basically the corresponding character that will be selected is basically digit or number will be selected numeric will be selected and then, it will display in the area. This is the text area where whatever the selection is there, it will be displayed; in addition to this numerix.

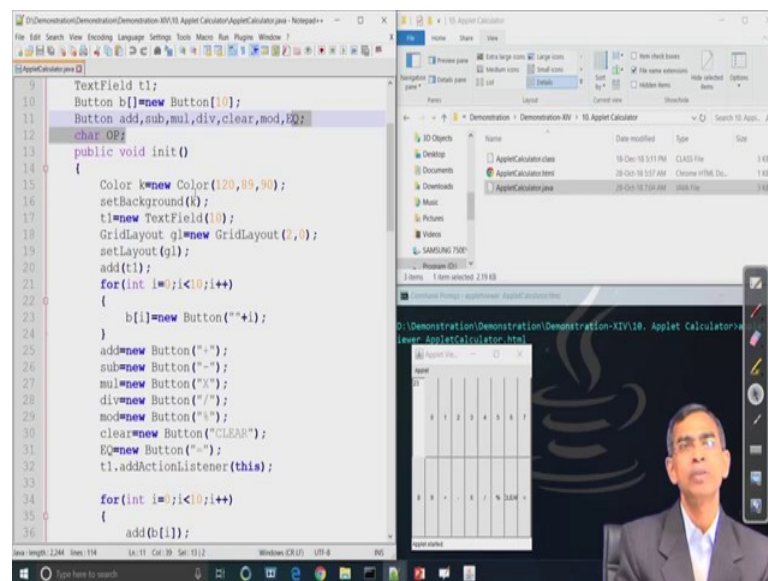
There are few more buttons we can see those are plus-minus, then star and then divide and finally, there is a button is called equal and then another clear. So, these are the few

buttons, this is a very simple one applet with altogether as you see total 16 right buttons are there those are arranged in that way and they are enabled, so that if user can click using mouse that particular event will be there in this applet in this background and they will sense it and then get it. So, it is basically an event handling concept is there although we do not know much about or on anything about event handling concept, we will just simply give it assuming that some events will occur there. If events will occur, they will be handled here.

Now, let us have the calculator program here. We will emphasize about only the design point of view and then, even handling point view we will be just simply mentioned here, but details event handling will be discussed in our another class.

Now, here we can see first of all we have to create the button as you see we have created total altogether 15 Buttons and then on another Button is not Button is the text fill area. Now, let us see how we have created 10 Buttons. First, we have created 10 Buttons here as you see here we can see the Button is the one class. It is there in AWT by which we can create the Buttons here. So, a list of Buttons is created here. We can see 10 Buttons are there and then, in addition to these 10 Buttons, other 5 Buttons that we have created they are named differently and then in the init method.

(Refer Slide Time: 20:42)

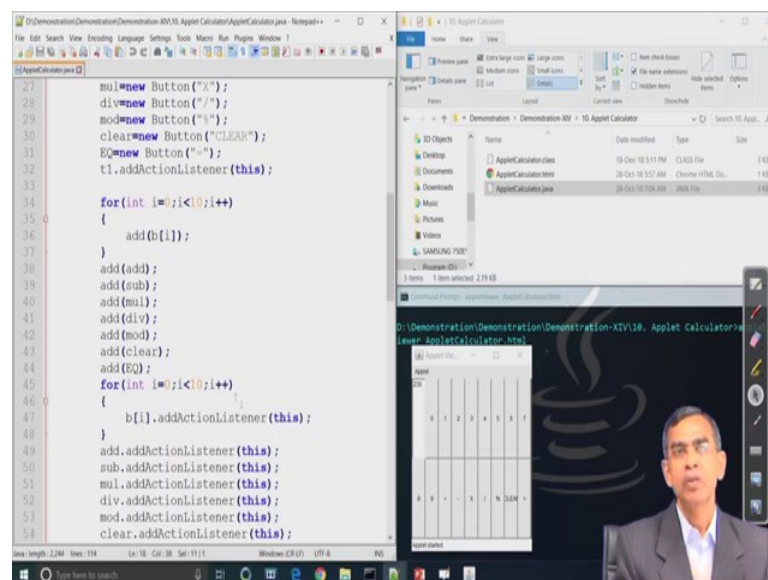


We just background color we know how to change the background. This is a Background Color of this one set Background and also we have created another area. This area is

called the TextField area. So, the TextField is another component and this is the size of the TextField means how many characters it can hold. So, 10 and then all these patterns and text fields are arranged in this arrangement. This arrangement is basically obtained by means of great layout pattern it is called all these classes like button the text field the great layout is defined in AWT package.

So, they will basically do it that all these buttons that you have created will be arranged in this grid form. So, these basically look like a grid and then, we once the button is created we have to add the button using add method there in the AWT class is there.

(Refer Slide Time: 21:41)



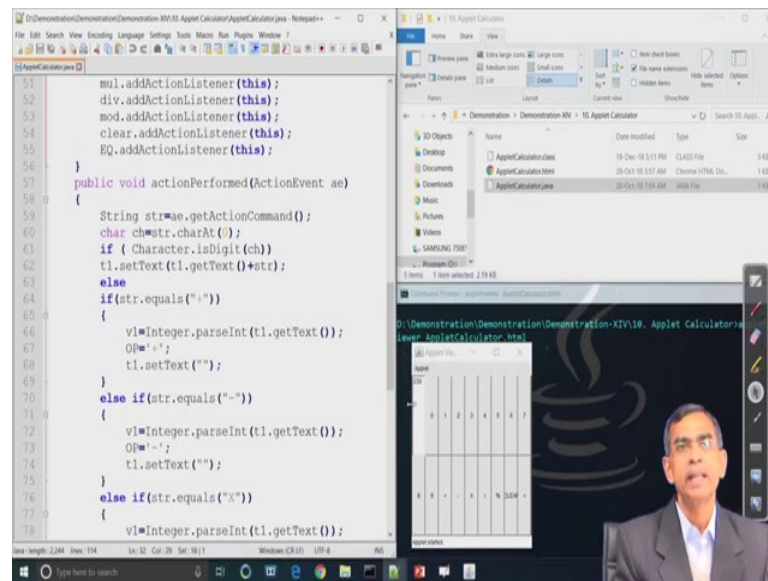
So, adding these all these buttons one by one into these grids and then, altogether all buttons are placed there now. So, this design completes all buttons are arranged, all text field is also created and next basically this basically look like, but here at the till this time we do not mention about any event handling concept. Now, in order to have the event handling concept what we have to do is, we have to register that a listener object will be there.

Now, this we do here in the start method by adding the action listener 1 method. Its interface is there in again AWT package it is. So, the action listener we have to include is there and this action listener you will basically once it is implemented and then, we have to include one method add action listener as a registration purpose. That means, action listener is an interface and add action listener is basically is a registration purpose that we

want to listen to this concept or element or the component, the button component and the other components are there.

So, for this so this is the mandatory one thing that we have to do it. So, this basically is basically starting point of handling your event and then, all these classes that we have declared here we have to add into this action listener here at and all these there. So, whenever we have already added all these buttons are there.

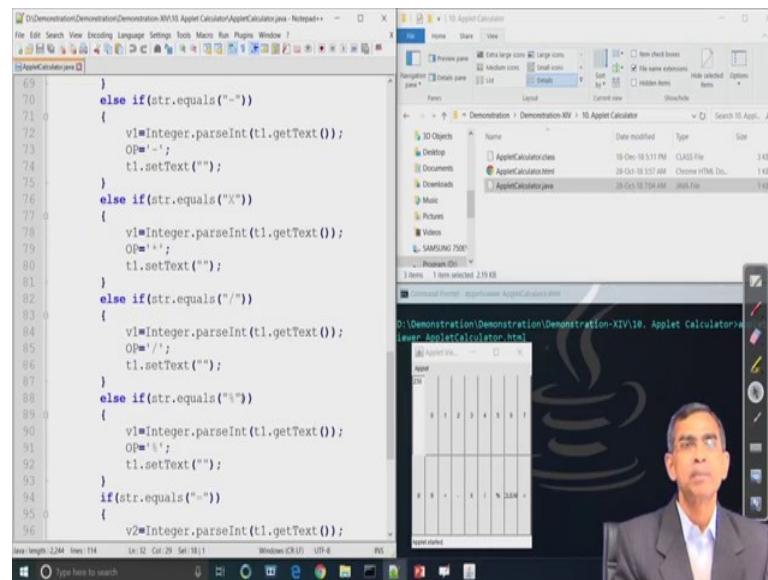
(Refer Slide Time: 23:20)



And then all these add action listener performance needs to be specified properly and here we do this. Here the get action commands are another method who basically gets exactly what is the character that you have to type while you click your mouse. So, it is basically mouse clicking and whenever you type it is basically passed as a string get takes and then, converted into the string to character or integer or whatever it is there and then, it gets the value.

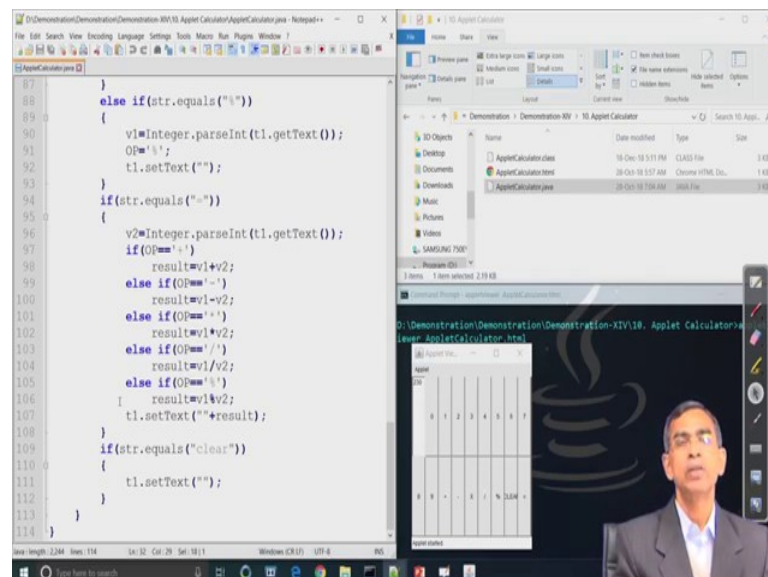
So, if we press it here get the text is 2 and then, it parts into the value as a number and in the number will be displayed in the text field area.

(Refer Slide Time: 24:00)



And these are basically how the event can be handled. That code is mentioned here is basically whenever a click action is there, it gets a text from there and then, store into a number and then, that will be displayed in the text field that is the event handling mechanism at the moment.

(Refer Slide Time: 24:16)



You may find a little bit difficult about dealing with this event handling mechanism is there nowhere for example, if this operation is plus. So far the action handling is concerned if we click plus, then it will work in this way and then by the resultant the plus

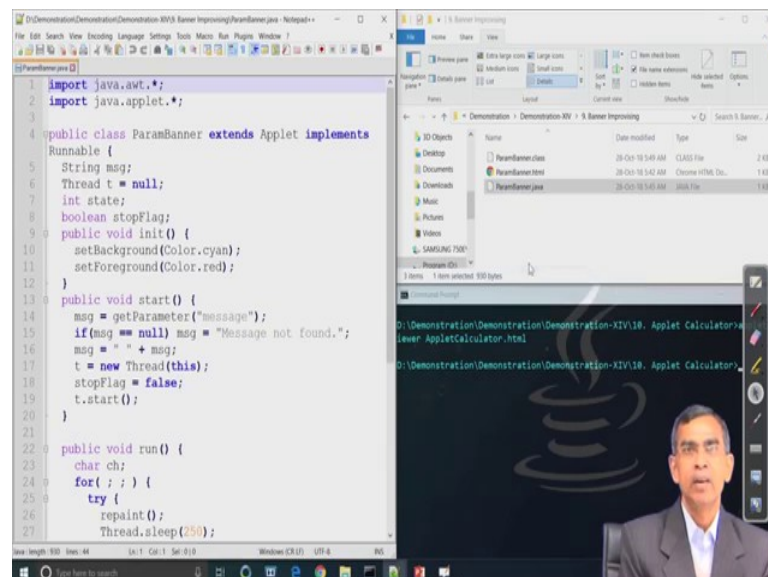
operation will take place. Anyway so, this way the applet contains few components they are called buttons and one component is called the text field area.

That is all about this applet content is concerned and for each applet components, there is an action to have listened and that action listening what we actually the action that will be to listen has been defined by means of the implementation of action listener methods are there. So, this is the applet code that is there and how it executed it is there.

Now it is as you see this is a very simple and then, very initial stage of an applet, but for this initial stage it is enough and there are many more things. The color can be changed, then size of each button can be configured properly so that it really looks like a good looking applet calculator it is although it is not at the moment.

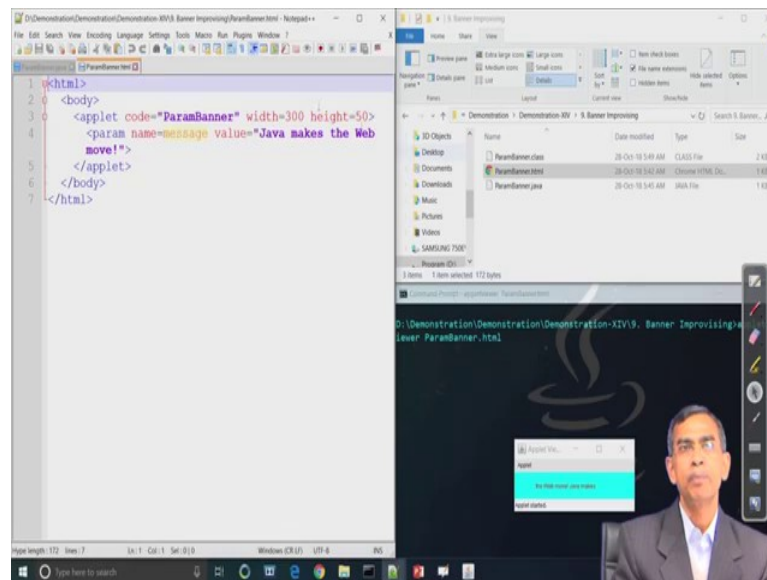
So, it can be done all those things will be discussed whenever we discuss about AWT programming followed by the event handling programming there. So, this is at the starting point for our applet programming and so far the advanced use of an applet concept is there. Now, another is the banner applets. We have a view of the banner applets or have already displayed it how the banner applet works there.

(Refer Slide Time: 26:12)



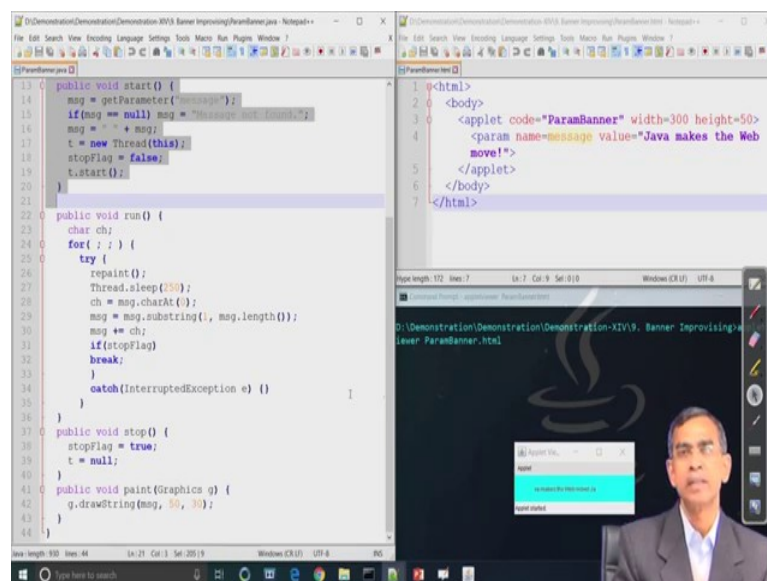
Now this banner applet that we have already discussed is basically a text which will be given from the banner the program class itself, but this is a little bit extended version of this one where the message will be passed through a parameter passing technique.

(Refer Slide Time: 26:34)



Now if we see the HTML file here, HTML file right and here you can see we want to pass the value to our the banner applet that we have already discussed earlier. Earlier the banner applet use the static message which is there in the code itself, but here we want to pass this value from the banner from the HTML page itself and the HTML page we will take this value and then, it will pass it to the applet and then, our banner applet we will run with the passed value it is there. Now, here is the demo we can see how it will work for us ok.

(Refer Slide Time: 27:18)



So, there is nothing as such details change about the compared to the previous one, the same concept banner everything only the message is there and then if there is no successful inputting from the HTML. So, a try-catch block is put that is the only extra part that we have added here, otherwise, everything remains same and as you see it will basically this applet banner we will run with the message java makes the wave move like this one and then, it is executed. So, it is another form of applet right ok.

So, the programming concept is there. So, we have learned about few more things right about how the applet tag is important. So, for hosting an applet code in HTML file is concerned, we have learned about it and then, we have discussed about the banner. It is an advanced version; a little bit in the sense that parameter can be passed from HTML and the very initial or beginning loop of a calculator applet. So, that is all for today.

Thank you very much.