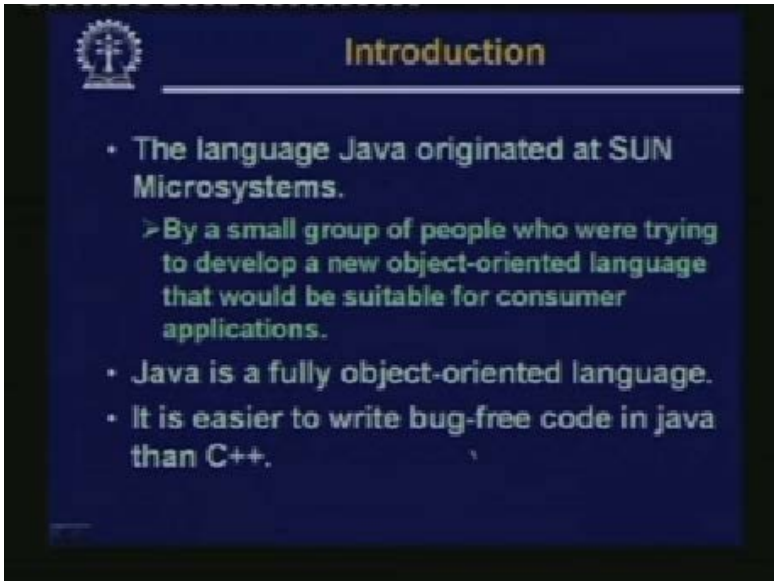
Internet Technology
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture No #28
Java – Applets Part: 1

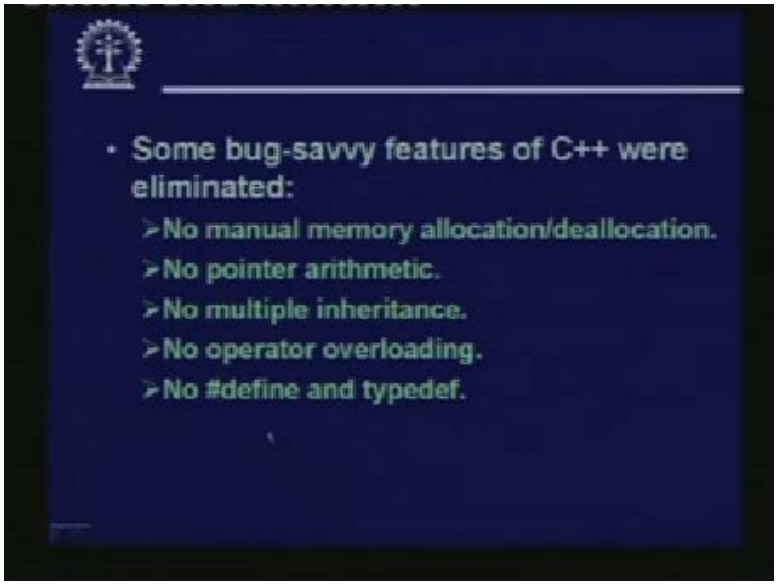
Today we shall start our discussion on Java Applets. Now you may know Java like Javascript is also an object-oriented language. In fact Java is a more elaborate language it is a complete language and you can use Java to develop any arbitrary kind of application however sophisticated it may be. Now in the context of internet we talk about Java applets. So we shall first be talking into the language Java in general terms then come to the point why we use Java applets and what are they? So Java applets this is our topic of discussion.

(Refer Slide Time: 01:22)



The language Java to talk about history first originated at SUN Microsystems by a small group of people who were carrying out braining storming sessions in trying to develop a new object-oriented language that primarily there aim was it should be suitable for consumer applications. They were developing very small and sophisticated instruments games consumer applications appliances where they wanted a programming language which would help them in configuring and programming such appliances in a much easier way. So the language Java originated in the process. Java is a full object-oriented language and it is also considered to be easier as compared of to C plus plus from the point of view of writing bug-free code. Why?

(Refer Slide Time: 02:34)

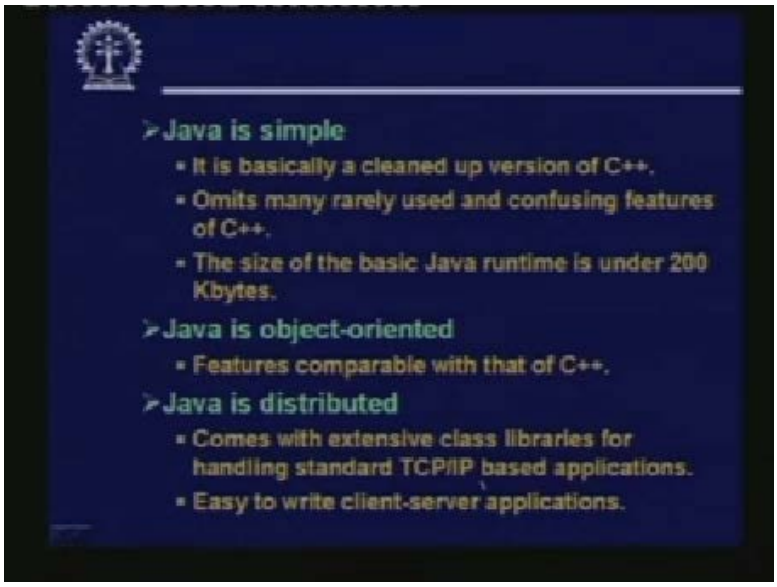


Some of these points we shall see here. The reason why that Java can be used to write bug-free code better than C plus plus or C is that some features of C plus plus or C which often lead to bugs in programs were eliminated from Java. Some of these are manual memory allocation de-allocation was removed. So a user would not be able to allocate memory or manually deallocate them which was considered to be a problem creating feature of C which frequently lead to segmentation false, these kinds of errors if there was some error in pointer manipulation. Java does not allow you to carry out any arithmetic on pointers. You use pointers in the way they should be. But do not add multiply subtract pointers. They are meaningless. Multiple inheritances are not allowed. This is considered to be a problem. Again operator overloading you cannot do. Of course you can the method overloading. But no operator overloading and no macro definition like hash define, typedef which can be confusing. So these features were eliminated from Java.

(Refer Slide Time: 04:00)

Java had some design goals. In the original published white paper for Java, the following design goals were specified. It was said that Java should be simple, should be object-oriented, should be distributed, robust, secure, architectural, neutral, portable and interpreted. Let us try to understand what these phrases or terms really mean.

(Refer Slide Time: 04:34)



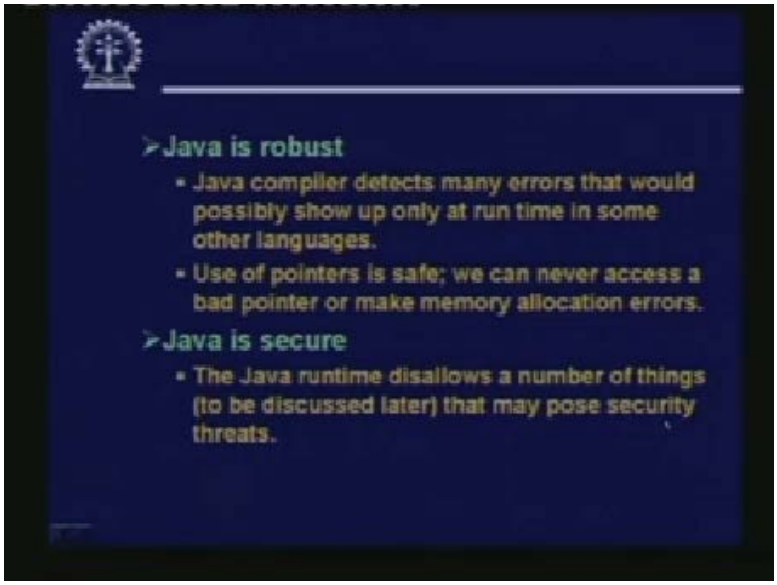
Java is simple. The basic concept was that the Java design white paper said that the language should be simple in design and simple to use. What it really means is that it is essentially a cleaned up version of C plus plus. C plus plus as an object-oriented language was already there. So Java is a cleaned up version of that. As I had mentioned few of the features which were eliminated rarely used and confusing features some of these which were present in C plus plus where omitted. And the language is simple you can appreciate it from this fact that the size of the basic Java interpreter is less than 200 kilo bytes only. So in terms of the resources that a Java program executing on a particular machine will consume.

This is also pretty limited, you do not require large memory to load and run Java programs. Even in embedded applications you only need 200 kilo bytes of memory to store the Java runtime of Java interpreter on top of which you can run the Java programs. Java is object-oriented this was one of design goals in fact Java is object-oriented. And the object-oriented features that exist in Java are quite compatible with that of C plus plus. Java is distributed, see Java one of the design goals was that it should be able to very easily write and develop distributed application because it was also considered that Java should be a language ideally suited for internet programming; programming on the internet.

So there are some features in Java ideally suitable for developing distributed applications. Like it comes with an extensive set of class libraries for handling standard TCP/IP communication. It is very easy to write client-server applications in Java. For those of you who had some prior knowledge in writing client-server application in language like C or C plus plus, you will know or you will see that the effort that you need to put into write similar application in Java is a greatly simplified one. In C you have to write a long piece of code for

communication in Java mainly 2 or 3 statements only to do the same thing. So this is the big advantage.

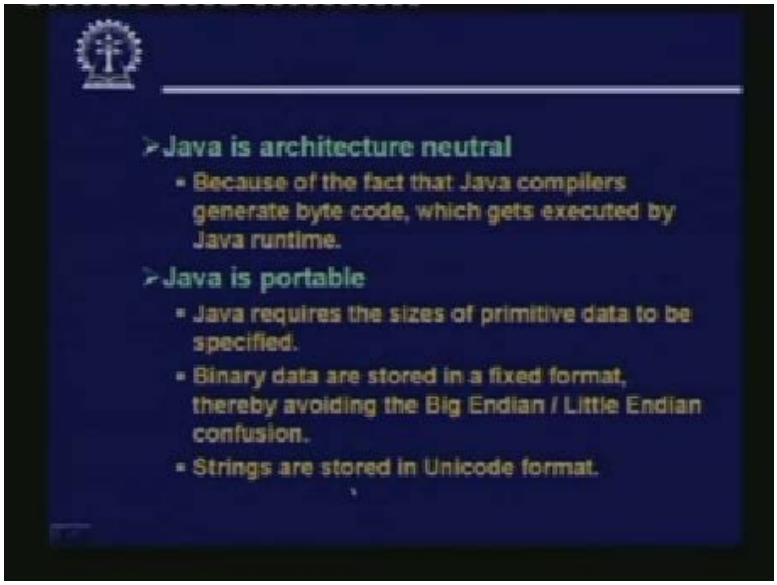
(Refer Slide Time: 07:33)



Java is robust. Robust because, Java compiler is able to detect many errors that would possibly show up during execution in some other language like C. Now one problem we often face when you write C or C plus plus program is that you write a program. Possibly there is some error in program logic. But the compiler does not give any error only during execution we get an error, segmentation fault and or some other kind of error. In Java many of this kind of errors will be detected during compilation only. This is one advantage and use of pointers is absolutely safe like in C where you can always be trying to use invalid pointers. If there is some error in a program in Java it simply not possible to access a bad pointer or make memory allocation errors. Because memory allocation is automatic.

There is no concept of manual memory allocation deallocation where you might make a mistake and secondly pointers you are not allowed to do peculiar manipulations of pointers like you cannot add pointers subtract pointers which in a sense are meaningless. But in C you can always do that without the compiler giving any kind of error. Java is secure. Java is secure because the Java interpreter or the Java runtime we shall see, this will not allow you to do a number of things that can pose security threats. This issue we shall we shall see in some detail later. But for the time being, you just understand or remember that the Java interpreter will not allow you to do the security vulnerable issues or the things which can pose as a security threat during execution.

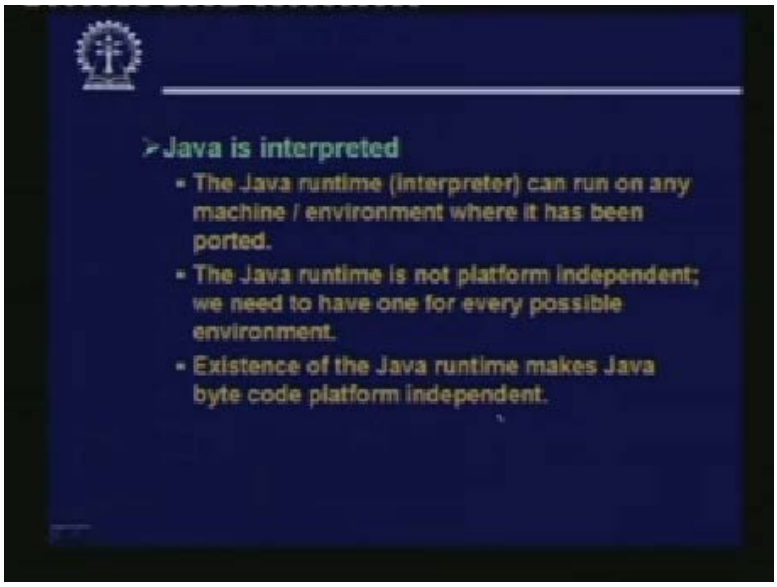
(Refer Slide Time: 09:48)



Java is architectural neutral. This means that the Java programs can run on any platform, on any architecture. This point again we shall explain later. This happens because the Java compilers generate a platform independent byte codes which are executed by the Java runtime. The language is considered to be portable because the features of the Java are very strongly typed. Whatever you use and specify in Java, there is no ambiguity anywhere. Like for the primitive data types. Like integers floating point Java requires the size to be clearly specified. Binary data are stored in a fixed format across all machines. This Big Endian and Little Endian confusion keeps haunting us. You know there are certain class of machines which use the Big Endian format.

Where the high order bytes are stored before the lower order bytes. But in Little Endian format it is the reverse way is the other way round. So when you are storing an integer or a floating point number in a memory, the order in which they store in the two machines will be totally different. So when you store this data or dump this data in a file, these files will not be portable across machines. But in Java this issue is not there because in Java there is a standard and all Java implementations follow the same standard and also strings. All strings in Java are stored in 16-bit Unicode format. So here also there is issue of standardization or you can say lack of standards. So it is all Unicode.

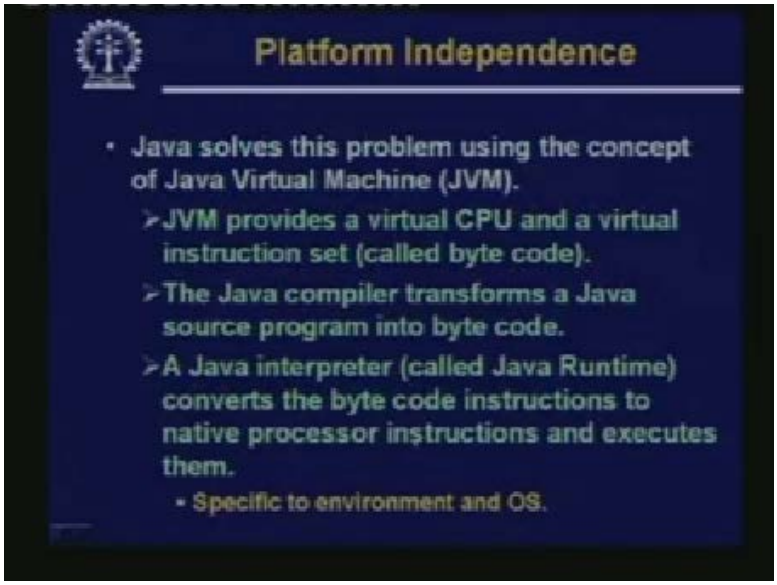
(Refer Slide Time: 11:50)



Java is an interpreted language. The Java runtime or the interpreter as we had called, this can run on any machine or environment where it has been ported. The Java runtime is not platform independent. This we will explain again. We need to have a runtime existing for every possible environmental platform like Windows, Linux, SUN, HP, etcetera. This Java runtime makes the Java byte code programs platform independent. We shall be explaining. So we shall be explaining this issue of platform independentness again in detail. Because this concept is very important. What makes Java programs platform independent?

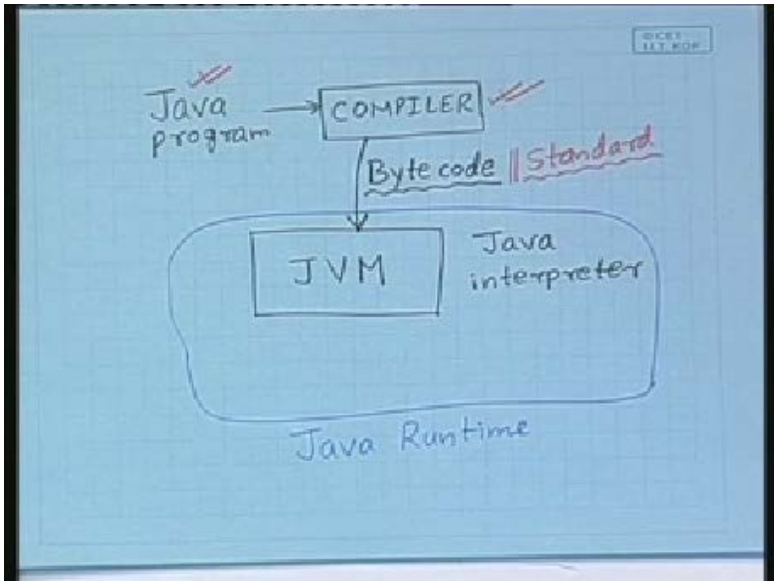
Why they can run across platforms they can be transported over internet? Because one thing you understand Java, one of design goals of it was to have a language using which you can develop internet applications or we can use it over the internet. Whenever we say internet, that means you are downloading or transferring transmitting Java files from one machine to another. So here the issue of platform independence and portability naturally comes in.

(Refer Slide Time: 13:21)



So some features of Java there which allows you to do this. Let us see how this platform independence feature actually comes into the picture. In Java there is a concept of a Java Virtual Machine. Now let us try to see what this Java Virtual Machine means.

(Refer Slide Time: 13:41)



Java Virtual Machine is basically a Java interpreter. This can interpret Java programs. But there is one difference between the interpreter that we had for the Javascript and the interpreter that we have for Java. In Javascript the interpreter can take the source code and can interpret directly. But in Java it is a little different. It is something like this suppose you have a Java program out here. The Java program you feed to the Java compiler. Now here again this com this compiler works in a slightly different way. Because the compilers that we are normally familiar with, they translate a high level language program into target machine

code which can be directly executed on the machine and because the target is the machine code of the machine of the particular computer where you are running.

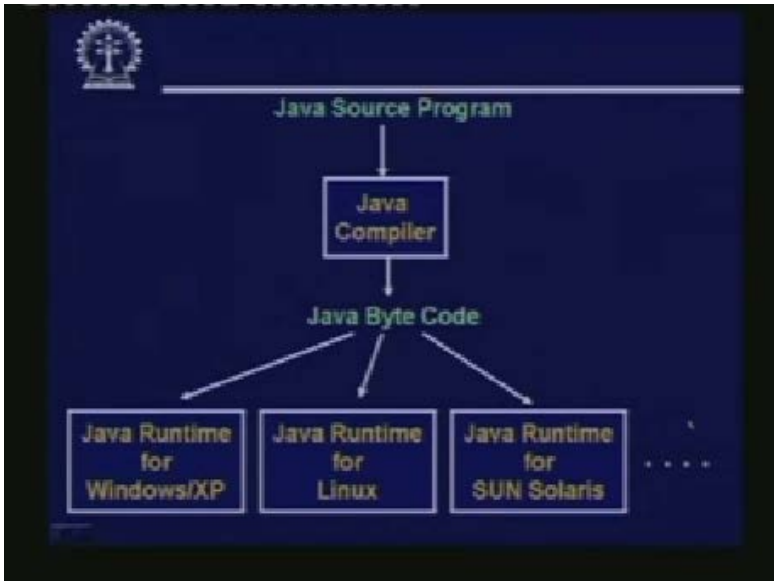
The machine codes are not platform independent. A program; a C program you will have to compile for the Linux environment, you will have to compiler separately for the windows environment, separately for the SUN machine and so on. But in Java what is done is something different. The Java compiler will be generating the object code in some intermediate format. This is called Byte code. Java Byte code. Now Java Byte code can be considered to be equivalent as it is the machine language of a hypothetical machine, which does not exist. But it is a simple enough machine for which the compiler can easily generate code. So Java byte code is basically considered equivalent to the machine language of a hypothetical machine. So that is where the concept of Virtual Machine comes in.

So as I had said the Java compiler transform a Java source program into byte code. Now the Java Virtual Machine that we are calling, that is sometimes also called Java runtime. This is basically the Java interpreter which will take the Java byte code and execute them directly or interpret them directly. Now let us again come back to this diagram. Now you see this Java program is something which you have written. This compilation you can do on any platform you want. What you know that this byte code that you generate this is a standard byte code. This is defined by the Java standard. This byte code will be platform independent because it is defined according to some Java standard where you compile. Whichever compiler generates the byte code, the byte code format will be standard.

You can easily take it from one machine to another without any problem. But the point to note is that what you have after that is Java Virtual Machine. This portion, now JVM is not platform independent. What is JVM? Java Virtual Machine as I had said this is also called Java Runtime. This is essentially a Java interpreter which can interpret Java byte codes. Now this Java interpreter must be actually running on the different computers on the different platforms. So you should have one version of Java Virtual Machine or Java runtime running on Linux, one version running on Windows XP, maybe another version running on Windows 2000, another version running on SUN Solaris, another version running on HP UX and so on. For all, the target platform you must have separate versions of Java runtime installed and running. Fortunately Java runtime is available on almost all platforms today.

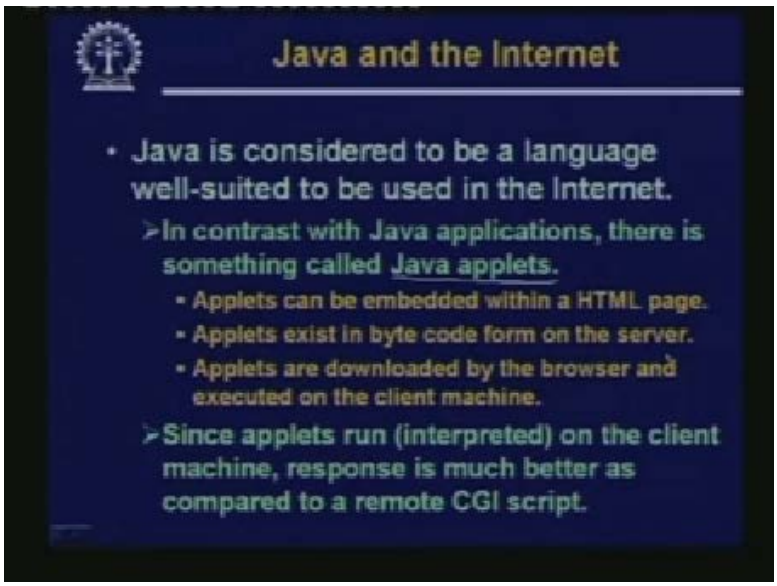
All you have to do is simply download and install Java runtime. Java runtime gets installed as an add-in to your browser also so that you can run Java program in your browser itself. So once you install this Java runtime on a machine, Java programs now become totally platform independent that can run on any machine. If a machine where you are downloading a Java program has the runtime. It can execute there without any problem and this is how platform independence is achieved there are two things. First thing is that the byte code which is platform independent and portable. Number 2. The Java runtime which is preinstalled on all the platforms. Java Virtual Machine is not portable. But the Java byte code is. This is how we manage or handle the problem and issue of portability of Java programs.

(Refer Slide Time: 19:58)



So this diagram which is shown here. Actually re actually iterates the same point I am trying to make. The Java compiler generating the byte code which can execute on any platform. It can be Java runtime on Windows, Java runtime on Linux, Java runtime for SUN Solaris or any other. So this is how Java programs get executed.

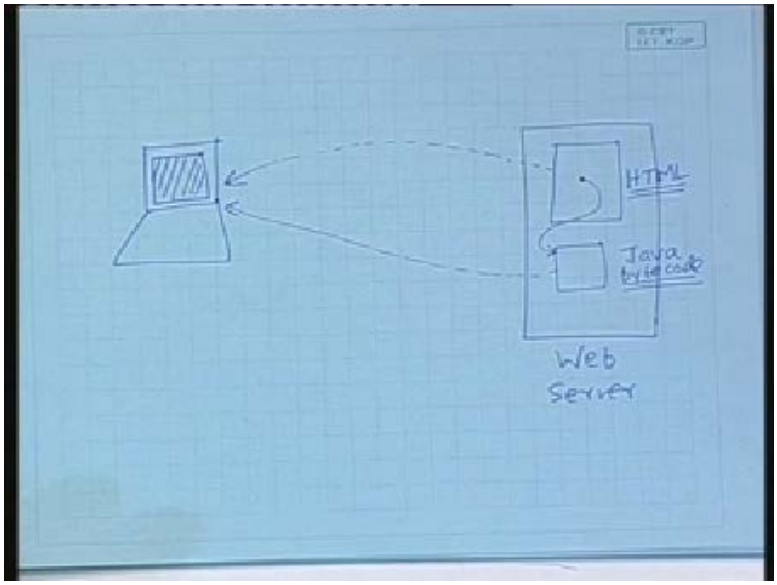
(Refer Slide Time: 20:28)



Now Java is a language I had mentioned which is considered to be very suitable to be used in the internet scenario. Let us try to understand here that quite little say so. Now for use in the internet, there is something called Java applets that we need to understand. What are Java applets? Normally we you write a Java program they are called applications Java applications. But Java programs to be used in the internet scenario as part of web pages. They are called Java applets. So basically Java applets are Java programs which can be embedded within an HTML page. Applets exist in byte code form on the server and applets are

downloaded by the browser executed on the client machine. So what I have said is this. Let us try to understand.

(Refer Slide Time: 21:39)

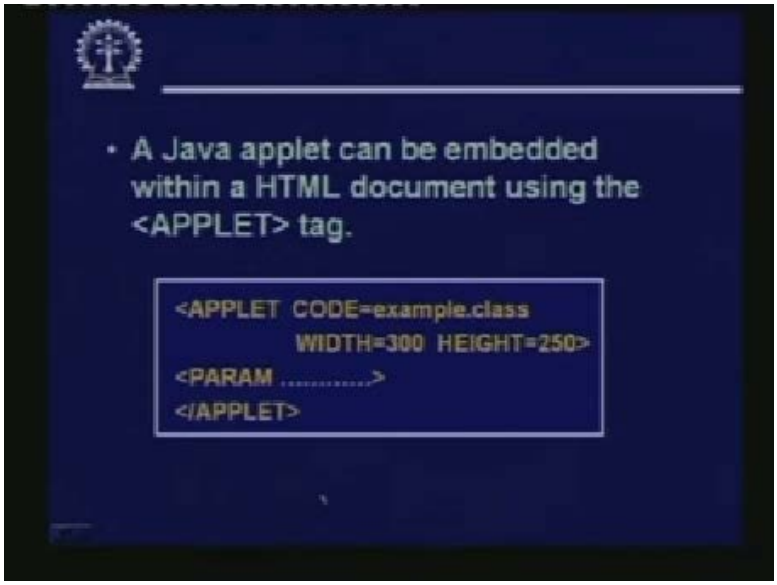


This is your web server. This is a web server on which we have a particular HTML page. This is an HTML page and there is also a Java program which is stored on the server. This is actually Java byte code after compilation and from this HTML file there is a link to this applet. We shall see how Java byte code, now from the browser suppose this is your browser from here you are giving request to download this web page. So when this web page will be downloaded and displayed on your screen following this pointer. This Java byte code will also get downloaded. So now what will happen is that we will be happening a scenario where you will have a page being displayed on a screen with a Java applet inside it.

So this Java applet can now execute within the browser itself within the web page itself. Now these Java applets can be used to provide highly interactive and animated web pages which make the web page design pretty interesting and attractive. Now if you compare Java applets being downloaded on the browser and executed locally versus CGI scripts. The advantage is that the response will be better because applets run locally on the client machine. Now again another question might bug you that well there are many thing I can also do using Javascript. I can also do using Java which one is better. Now here there is a trade-off. If the thing you do u want to do is simple enough it involves few line of codes possibly Javascript is better.

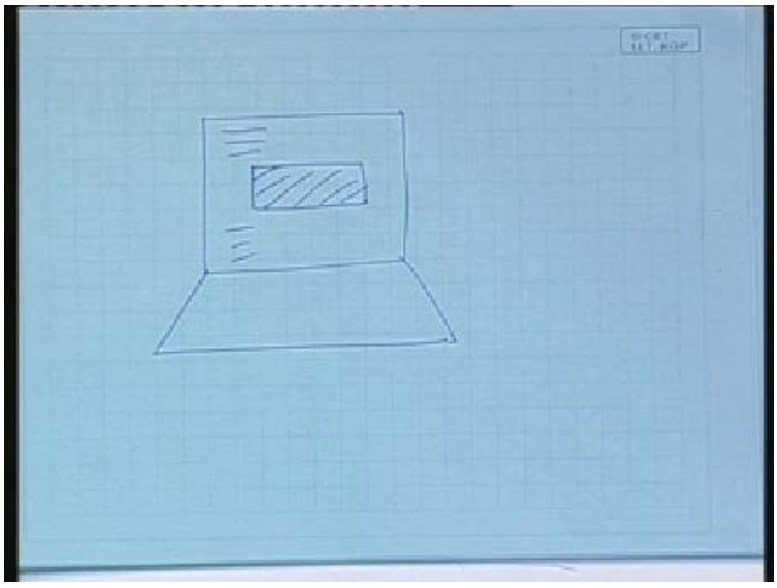
Because you will have to download Javascript or Java whatever from the web server to the web client. If it is embedded Javascript you need to download the file only once. But if it is a complex application, like a very complicated animation which demonstrate the functioning of some protocol. For example graphically you can see on your screen how the TCP protocol functions. This kind of a thing it is much better and natural to have it coded in a Java program. Because in Javascript you simply cannot do this and put it as an applet that will be downloaded along with the page and you can see the animation playing on a screen. So the response will be much faster in general in Java programs.

(Refer Slide Time: 24:59)



This is how you can embed a Java applet within an HTML document. You use the APPLET tag, the syntax is this. There are some other attributes also but these are the minimum attributes you normally use; the first attribute says CODE. CODE gives the name of the byte code file by default the byte code file have an extension of a dot class. Java programs have an extension of dot Java. When you compile them you get a dot class file that is the byte code. So you specify the name of the byte code file to loaded, you specify the WIDTH and HEIGHT of a sub window inside a window.

(Refer Slide Time: 25:57)



That means, this is your window on the browser and you can specify here is some HTML document being displayed. You can specify in between that this is an area where I want my applet to be executed. So I must specify the size of the display area in terms of the WIDTH and HEIGHT this can be specified as part of the APPLET tag. I can give WIDTH equal to

300, HEIGHT equal to 50. For example there are in pixels. An APPLET tag will end by the end APPLET tag. In between I can have one or more parameters statements PARAM through which we can pass some parameters to an applet from the HTML file. We shall see some examples later.

(Refer Slide Time: 26:51)

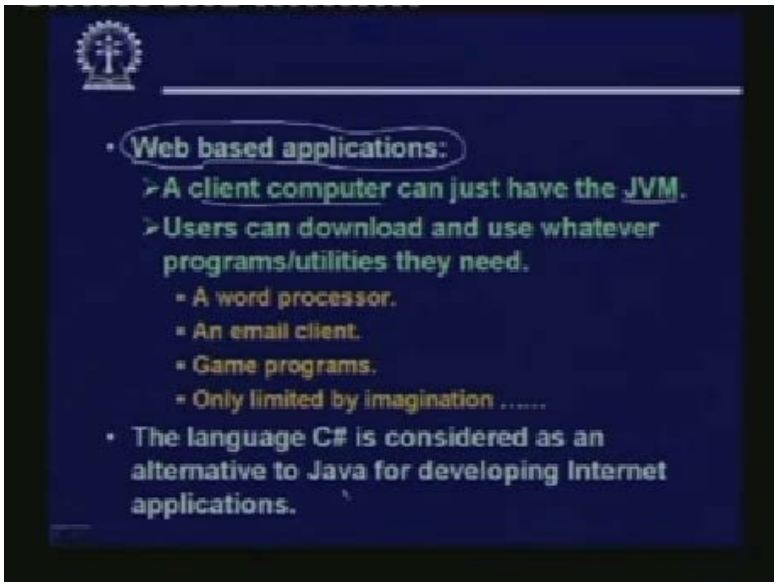
Now the question arises why do we need applets? Isn't? Java applications which I had talked about earlier are it not sufficient? Can't we do all these things using applications? We do. We need a separate kind of Java programs called applets there are reasons. The first thing is that there is no doubt that Java is very convenient for graphical programming. There are a large number of libraries utilities and methods available for programming involving graphics interactive applications and also network programming. All these three things can prove to be very attractive for developing applications in the internet scenario. Because in the internet scenario you actually have to deal with all these things. You need to make your pages attractive you need to have a very efficient network programming to transmit data.

Suppose you think of an application typically application you are developing some multi party online game using Java. There are a number of people which are spread all around the globe. They are actually playing among themselves. But during the time each of the players is making a move the data corresponding to the move has to do some central server. From there some response again has to come back. So there is continuous communications that need to go on between the client and some central server. So for these purposes network programming also becomes very essential. These are the capabilities of Java and we actually require and need Java. To do all these things, we need applets also because applets are something which is different from Java applications.

Java applets. This concept allows us to download Java programs over the internet and run them on the browsers themselves. Java applications cannot run directly on the browsers. Java applets can run on the browser. Because for Java program to be run on the browsers there are several things to be handled. Like I can move forward backwards scroll up down. So what happens to my Java program running, but when you are running a Java application it is only

that Java program you are running nothing else. But in a web page there are so many other things. There can multiple Java applets. There is an HTML document to be displayed. So many things are together. So, how to handle it in a browser? These are some issues to be addressed and applets as you will see due to their features you can use them to design an attractive or interactive web pages.

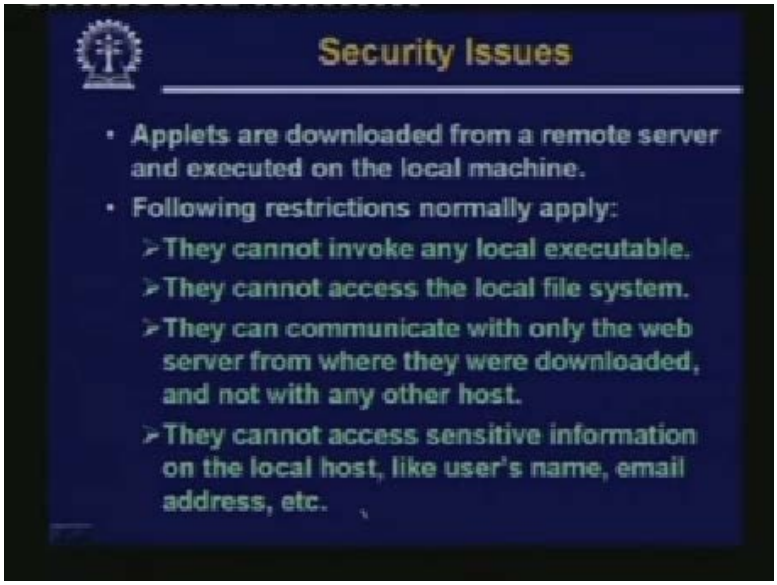
(Refer Slide Time: 30:04)



Now using Java applets you can develop web based applications. Now why web based applications? Well I do not try to limit your imagination. It is up to imagination. What are kind of application you would like to develop on the web? Now I can just mention a few of the interesting applications. The client computer from where you are accessing the internet can have just the Java Virtual Machine. It can be your computer; it can be your small handheld iPod kind of device which has of the capability of connecting to the internet and download Java. Java applets in fact some of today's mobile phones also have the capability of downloading Java applets. They can execute directly on the mobile device. So you think of a situation where in the client computer you have only the Java Virtual Machine installed. And users depending on their requirement can download any application or utility they want and use them.

This application can be a word processor it can be an email client, it can be a game, it can be whatever. Now in this kind of a scenario you need not install any special software or programs on your computer. There are many such machines which have been developed or built based on this principle. The machine has minimal configuration in terms of both hardware and software. Whatever you need you simply download from some server and run them. This is the basic concept. So these are some advantages which Java applets provide subsequently Microsoft has also come up with. You can say an answer to Java; a language called C sharp which looks somewhat very similar to Java in terms its capabilities. C sharp like Java can also be used for developing internet applications pretty easily.

(Refer Slide Time: 32:32)



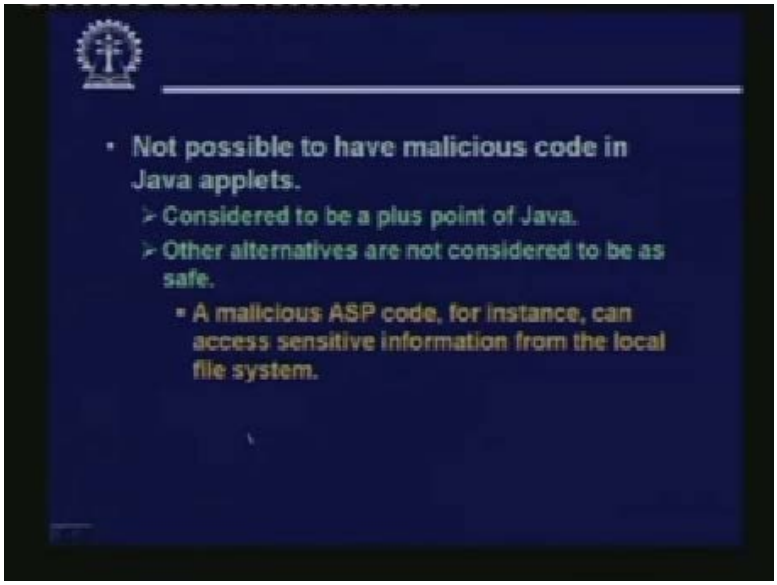
Security issues become very important here. You will see we are talking about Java applets. We are talking about Java applets being downloaded from some web server into my own machine. I do not know what that applet is doing. But I am worried about the security of my machine. I would not like that some virus goes into my machine. I would not like that someone can extract some sensitive information from my machine from my file system. So when I am downloading an alien code, a code which I have not written someone else had written. I am downloading that on my machine and I am executing it I must be absolutely sure that executing that alien piece of code will not do any harm to my machine. This is very important to understand and assess.

So the main security issues as I have said arises from the fact that applets are downloaded from a remote server and executed locally. There are some restrictions which applies when applets are executed which automatically ensures that security is not tampered with or is not breached. An applet cannot call any local executable file. This is not allowed. An applet cannot read or write into the local file system. This is not allowed. It can communicate with only the web server from where they are downloaded and not with any other host. What this means is that, suppose I am visiting a web site xyz.com, I am downloading an HTML page along with an applet from there; the applet was residing in xyz.com. Now inside the applet there is some network programming it is contacting some other program somewhere and is communicating some data.

Now what the constrain say is that this applet can only initiate a communication from that xyz com web site from where it was initially downloaded it cannot start a communication with some other web site abc.com. So these are some constrains. So due to these constrains obviously an applet cannot access sensitive information from the local machine. Like in some local machine in some special file the users name email address. All these things might be get stored. So if a malicious applet can access this information. They can send all this information to some central place. From there they can get some information which otherwise you not have like to disclose to others. But due to the restricted environment under which

applet runs due to the capabilities of the Java runtime these security issues are not that important.

(Refer Slide Time: 35:57)

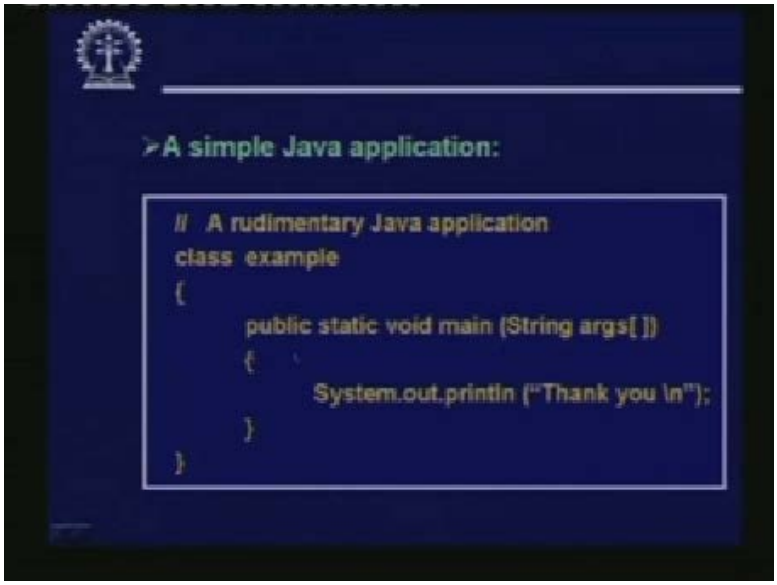


So you cannot have malicious code in Java applets. You cannot access any memory location outside your area. These are ensured automatically because Java is always executed in the interpretive mode. The Java runtime always has the control and it executes the Java byte code instruction by instruction. So this is considered to be a plus point for Java. But other alternatives you people use like ASP or some other technologies. There it is possible to write malicious code rather easily. So whenever you are installing some antivirus or some kind of a personal firewall you have to make a lot of configuration there. We check for these kinds of contents to ensure that they do not have or contain any malicious content. But in Java you can be absolutely sure of a Java program can never contain anything which can be harmful to your computer.

(Refer Slide Time: 37:03)

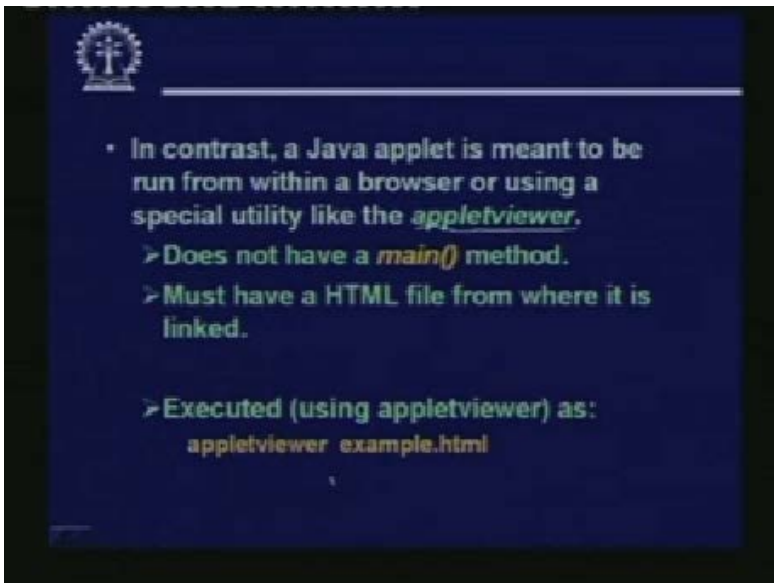
Now let us come to the point. Java applications and applets. What are the exact differences from these two? Well we shall see later more formally that how an application can be converted into an applet but simplistically speaking Java application is a standard alone program. They are standalone programs just like any native program native program means say I write a program is C. I write a program in C plus plus. I compile them. I execute them. They are standalone programs which are running on my machine. Java application work in some more similar way. You can compile a Java program as javac which invokes the compile. An example, Java is the name of the Java source file and after compilation you get this example dot class which is the byte code file. To execute the Java program you have to invoke the Java runtime whose name is just java. You call java with the byte code file the program gets executed. As I had said the Java byte code file is portable you can simply copy this Java byte code example dot class to any machine any platform and you can execute. There the program will execute correctly. So a typical Java application looks like this.

(Refer Slide Time: 38:44)



This is the simplest possible Java application which simply prints a line of text on the screen Thank you followed by a newline. There is class called example and within that class there is a single method called main. For a Java application, there has to be a method called main; this is mandatory. But we will see here there is a little difference from Java applet. In a Java application you must have method called main.

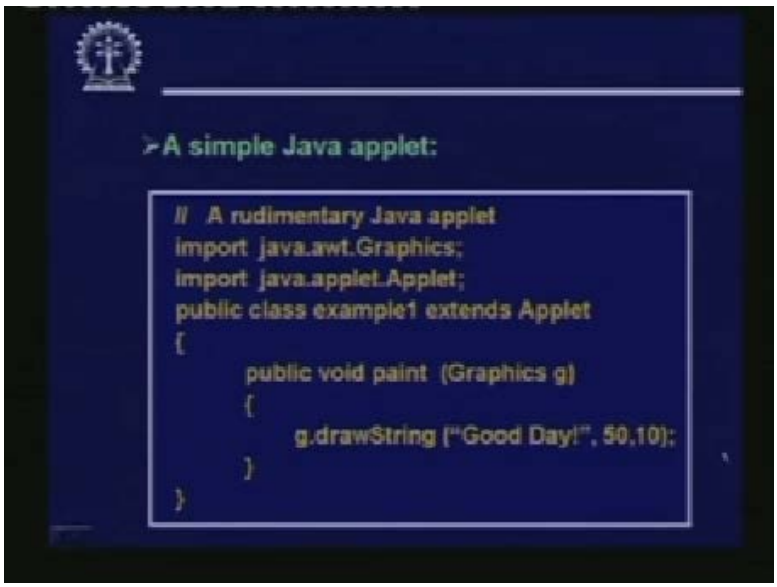
(Refer Slide Time: 39:23)



Java applet in contrast is not meant to be executed in a standalone fashion just like in the previous [39:33 word not clear] in the executing, a Java byte code by giving the Java byte code file this kind of a command. Java applet are meant to be downloaded along with a page web page and executed along with it. Of course for testing out applets are written together. There is an application which was developed by SUN again called applet viewer for running specifically applets on a window. Now some differences are there in a Java applet as

compared to Java application. This does not have main method and it must have an associated HTML file from where it has to be linked. And if you want to execute a Java applet say from an applet viewer you must call it by the HTML file name; not by the dot class file; not by the byte code. So applet viewer is actually a utility to view the complete HTML document along with the applet; not just the applet. So you need to invoke the HTML file with applet viewer. Similar thing you will do when you are including the applet inside an HTML document.

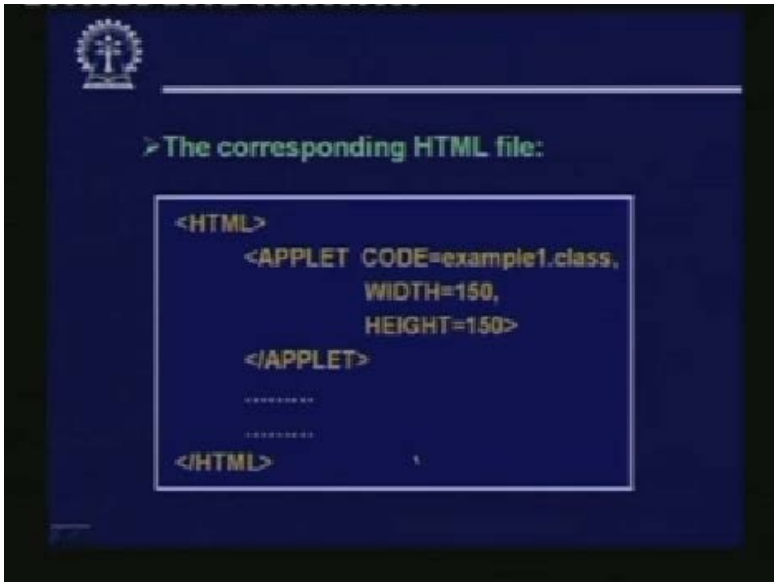
(Refer Slide Time: 41:00)



Now a simple Java applet will look like this. Now we shall see these details later how to convert an application into an applet in our next lecture. At the beginning there are some special libraries to be included. So as to have applet and if you compare this with the previous example here. Here we had a class simple and we had main function main method. Here we had also, here you have also a class whose name we have given as example one. This is this class is inherited from the master class applet and this does not have any method called main or rather it has a method called paint. There are some standard methods available in applet like [41:50 word not clear] paint redraw.

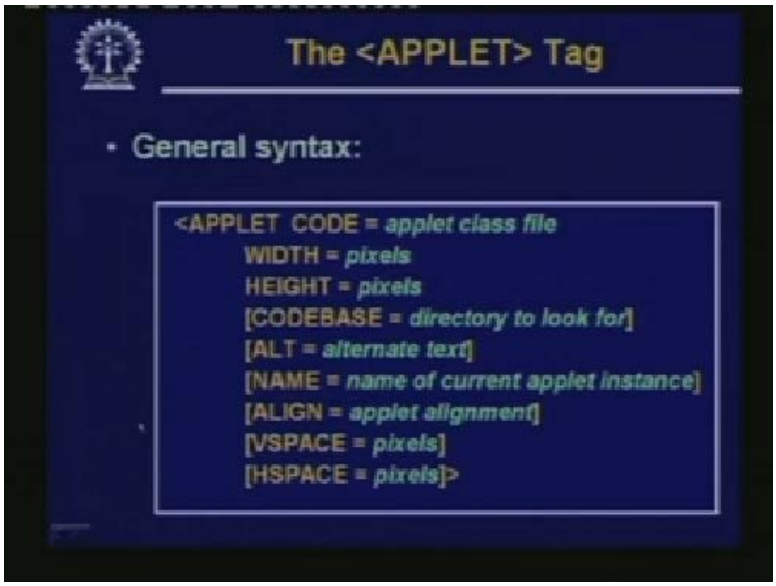
We shall see these later. This takes as a parameter a Graphics object g because in applet whatever screen area is available to you that is, considered to be a graphic object you can display anything on that graphic object. So draw string is a method which is present in the graphic object g.drawString. You can call this to display this string and this represents the actual coordinate where you are displaying. So in this way you can write a simple Java applet which is somewhat similar in capability to the application. We had just seen and to include this particular applet from HTML file.

(Refer Slide Time: 42:46)



The corresponding HTML file will look like this. There is an HTML where in addition to an HTML code, you have the APPLET tag. Code will example one.class. WIDTH and HEIGHT are specified of the window where the applet will be running one thing you remember here. That when you write a Java program, the master class. Here the name is example one the name of the Java program has to be example one.java and name of the byte code will be example one.class. This is the convention which is followed the same name of the class we use as the name of the file. So this is how the HTML file looks like.

(Refer Slide Time: 43:42)



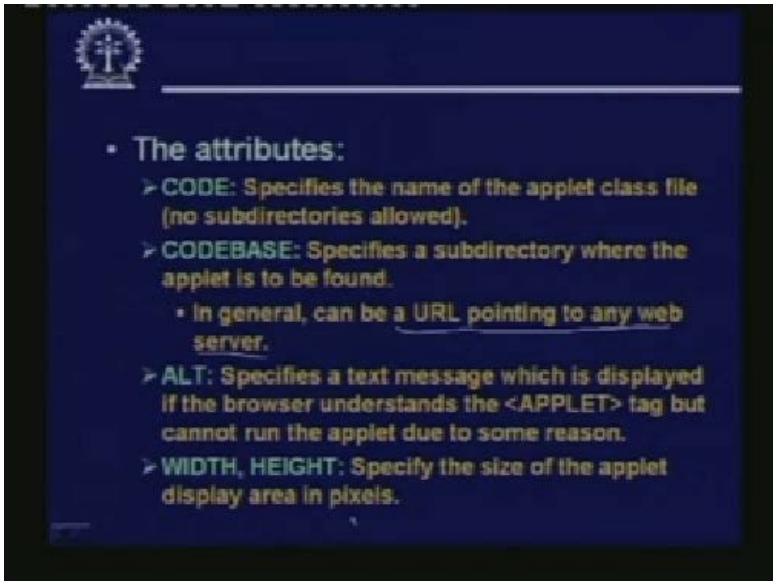
Now let us look at the general syntax of the APPLET tag. There are actually so many fields and attributes here. So let us try to see what these attributes are. APPLET CODE we have already seen there are other attributes. We will explain these three purposes WIDTH, HEIGHT, CODEBASE. These are all optional all fields within square brackets are optional

fields CODEBASE, ALT, NAME, ALIGN, VSPACE, HSPACE. These are all parameters, some of CODE, WIDTH and HEIGHT are mandatory; others are optional.

(Refer Slide Time: 44:24)

After that you can have as many number of parameters as you want. The syntax is PARAM NAME equal to name of the parameter; VALUE equal to value of parameter. This we will see an example for this later and after this you can have an HTML. Some HTML you put here because you know all browsers may not be supporting Java. If it supports Java, if it has the Java runtime installed, then the applet can be download and executed. But if your browser does not have Java, then there is no point in downloading applet. So in that case whatever HTML statements or code lines you have after this PARAM they will be displayed as it is in place of the applet.

(Refer Slide Time: 45:18)

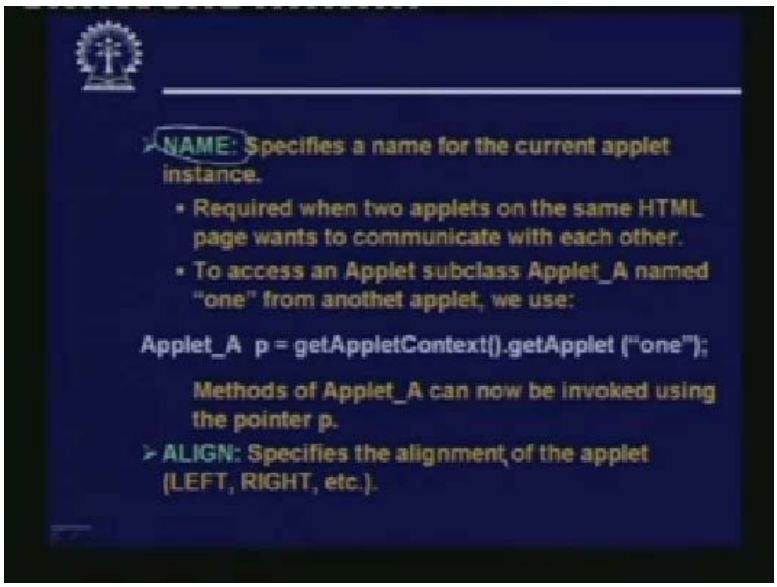


Now let us see what the different attributes of the APPLET tags specify. CODE we have already seen in the example. It specifies the name of the applet class file. Now when you specifying the applet class files subdirectories are not allowed. Like you cannot write xyz slash example one.class. You will only have to write example one dot class subdirectory specifications are not allowed here. But how do you specify subdirectories? If they are present, you can specify them by using the other attribute called CODEBASE. CODEBASE can be used to specify the default directory where the applet is to be found. So in CODEBASE you can write CODEBASE equal to xyz slash abc slash whatever.

Where the applet is to be found and CODE will actually contain just the name of the byte code or class file right. So in general this CODEBASE can be a URL pointing to any web server. So here there is a point you should remember that I am downloading an HTML file from a particular web server. In that HTML file CODEBASE points to some other web server. No problem I am allowed to download Java applet from some other web server. But the only constraint I have here is that the Java applets that I have downloaded can communicate only with the web server from where it is downloaded. Not from the web server from where I had got the HTML file.

So some Java applet which is stored in some other site I can also include them in my program, my code HTML document. So in CODEBASE you can have a URL which can point to any server in general. ALT is an attribute using which you can specify a text message. Here this attribute is used for a purpose like this. Suppose the browser understands applet. But due to some reasons it is not able to execute. Due to insufficient memory some reason. So whatever text is present in the ALT that text will be displayed in place of the applet WIDTH, HEIGHT are mandatory fields. They specify the size the applet window and the applet display and there are specified in pixels.

(Refer Slide Time: 48:03)

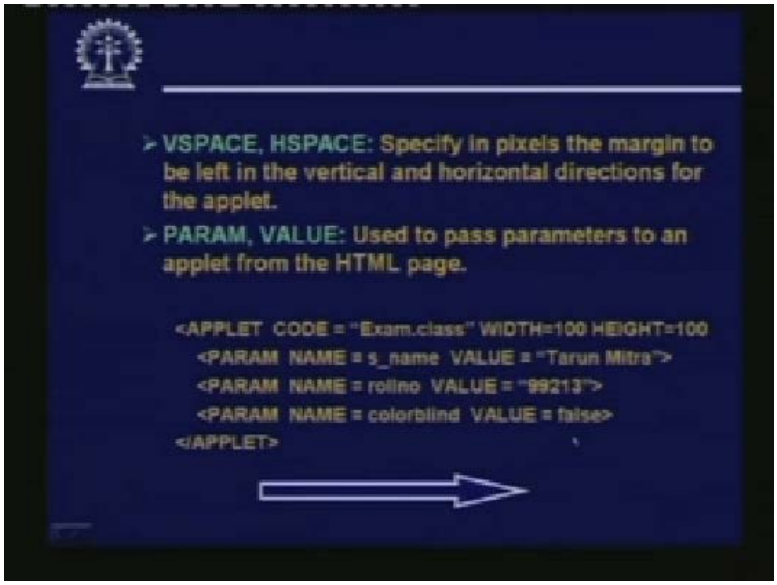


NAME sometimes you need to specify the name of an applet. Now when you were discussing Javascript in our previous class we had mentioned that when you want to call a Java applet

from within Javascript you need to include that Java class object byte's name. It is this NAME attribute which you should use for the purpose in the APPLET tag. Whatever name you have used this name we will have to use for referring to that particular applet. Moreover you can use this name for inter applet communication. Two or more applets on the same page HTML page can communicate with each other. For example if you want to access an applet subclass Applet underscore A whose name I have given as o n e one.

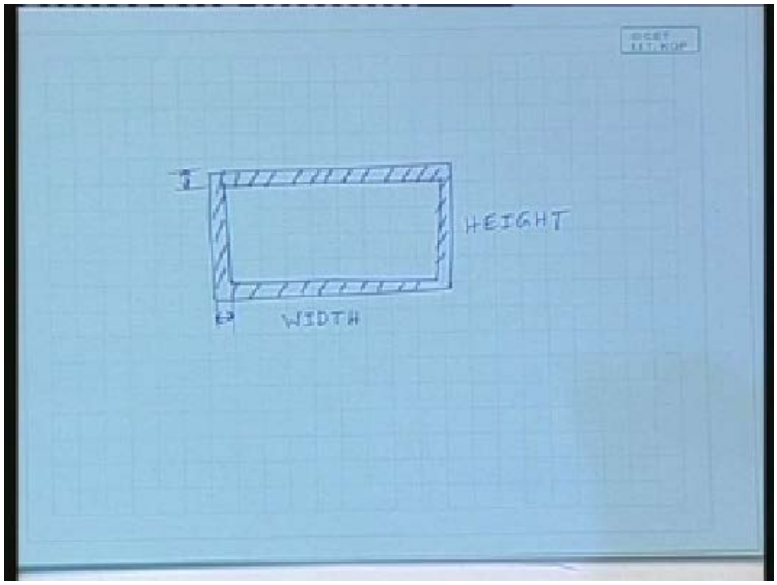
This NAME equal to one. From some other applet on the same page we can write a statement like this getAppletContext.getApplet within bracket NAME of the applet. So here we can get hold of the applet which we are wanting to access and assigning it to a variable p of type Applet underscore A. Now using this p we can access any public methods or public attributes of this applet one. This ALIGN this specifies the alignment of the applet. See we have specified the applet window. But we have not specified whether the window has to be centred. It has to right justified left justified. All these things all these things you specify by the ALIGN keyword.

(Refer Slide Time: 50:05)



VSPACE, HSPACE; this specifies the margin. See it is like this.

(Refer Slide Time: 50:13)



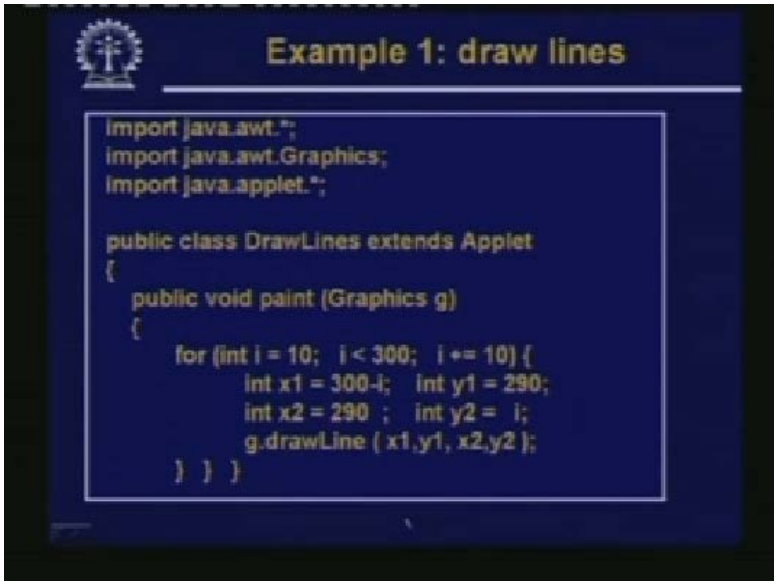
You have specified the size of the applet window. This is the WIDTH and HEIGHT. This is the WIDTH, this is the HEIGHT. In addition what you can specify you can specify some margin which has to be kept on the four sides. So this margin you can specify margin in the horizontal direction. How much margin you keep, similarly in vertical direction, horizontal direction, both you can specify. These can be specified by using this VSPACE and HSPACE attributes. PARAM VALUE I had said we can use this to pass some values to an applet. Let see through an example how we can do this.

Suppose in the HTML page you have included an applet like this. APPLET tag CODE equal to Exam.class. This is the name of the byte code file. WIDTH 100 HEIGHT 100 and there are three we are passing. Say, the names of the three parameters we are giving as sname or student name. Short form of student name sname, rollno, colorblind, whether the person is colorblind or not. And you are also passing the value for the s name you are passing value as Tarun Mitra. For rollno we are passing the value as 99213, for colorblind you are passing the value as false. So there are 3 parameters we have passed where you have specified the names as well as the corresponding values.

(Refer Slide Time: 52:17)

Now from within the Java program, this is some statement which are present inside the Java program. Now this PARAM NAME VALUE whatever said does a those are present in the HTML file from where the Java program is linked now inside the Java program you can have statements like this. getParameter s name, you assign it to a variable name, so actually that Tarun Mitra will get assigned to NAME. The rollno you again doing a getParameter rollno. You will be getting a retrieving a rollno value here. This will be obtained as a character string. So in addition you have to call integer.parseInt to convert it to an integer assign it to roll. Similarly again colorblind is the third field. You will be getting the value false getParameter Boolean.valueOf. If you call this, this will be converted into a Boolean value which we assign. So here you see the simple code snippet in Java where you can access the parameter values passed through HTML and you can assign them to some Java variables which you can use in whatever way you want after that. Now let us see very quickly look at some applet examples. We shall continue some examples in our next class also.

(Refer Slide Time: 53:56)



This is a very simple example which will draw some lines. Public class, this is the name of the class and as you can see here. Also we have a single method called paint. So whenever an applet runs it, this paint method which gets executed automatically. So by now you should be able to understand this. And in this for loop the values of i's are changing. And you are changing the value of this x and y coordinates. And we are drawing some lines so we will be getting a set of lines on this plane. So is just an illustration of drawing lines.

(Refer Slide Time: 54:41)

This another example which shows you how you can play a sound clip in Java. So these are the headers public class PlaySound. Here you have written a function, a method called init. There is a try catch block which is standard in Java catch is the Exception handler, the try block. You are creating an URL, new URL. You are assigning it to a variable called sound which is also of type URL. And a here in catch you call the Exception handler with the type

of the ex, type of Exception. If the URL is not properly formed and after you get, this you simply play sound. So whatever URL is there, that will be played and this sound you can hear. So play is a method which is built in.

(Refer Slide Time: 55:50)

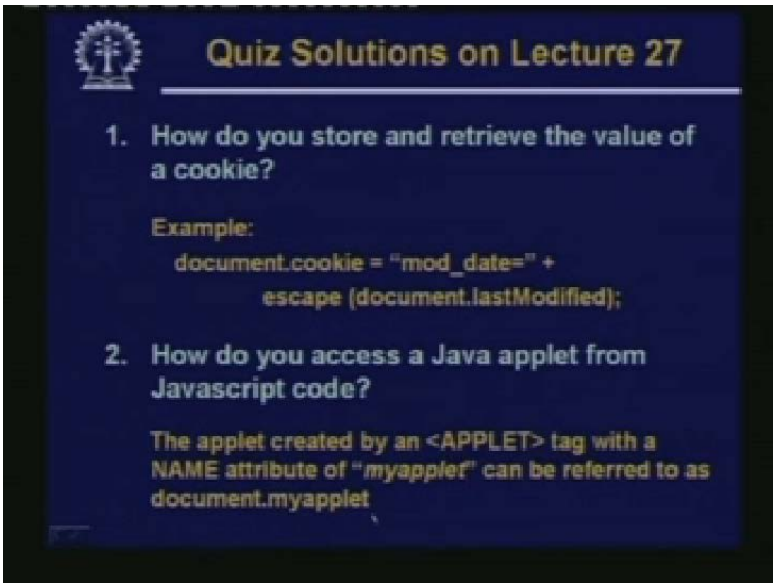
Display image again, this is another example. So here it is similar image picture null. Again you have the init function, again in URL you store a URL slash logo.gif and you can use the function get image. So display an image. This you can store in picture. So there some of examples of some applets. We shall see later some complete examples.

(Refer Slide Time: 56:20)

How we can use these? This imageUpdate paint all these things will be there part of it. So I am not going into detail of these. Right now this we shall come back to this later when we talk about examples on applets. What are the different methods in applets? Why are they

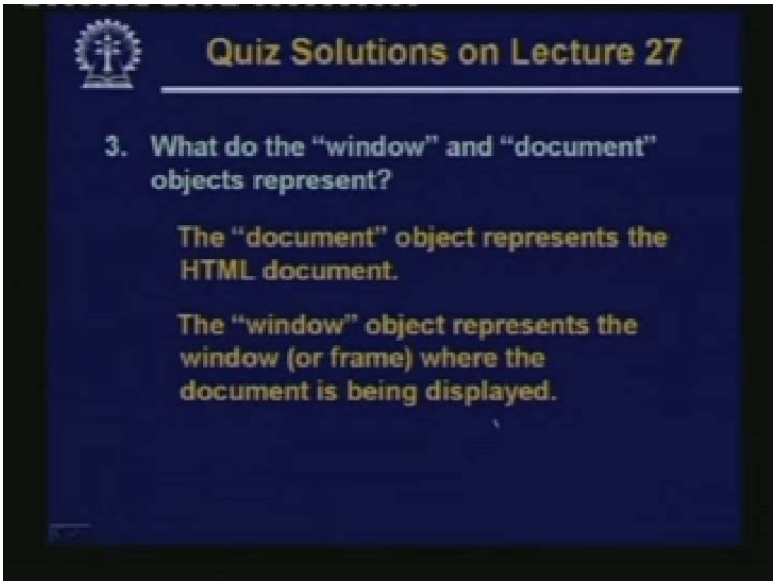
used? And so on. So let us stop our discussion on Java applets. Today here because in the next class we shall be talking about how we can convert applications into applets. Only then you will be able appreciate what are the different methods that are present in a Java applet and in what context they are invoked and used. So I am leaving the explanation for those detail functions invocation there order for the next class before that. Let us look very quickly at the solutions to some of the questions we had raised for the previous lecture.

(Refer Slide Time: 57:21)



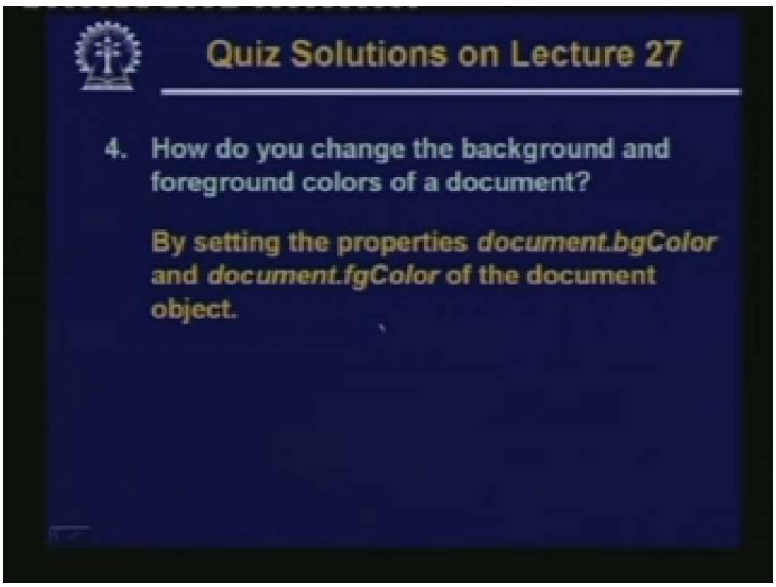
The first question was how do you store and retrieve the value of a cookie?
So the example we had given we can assign string to a document.cookie attribute and the value assigned should be of this form cookie name equal to the cookie value. This escape document lastModified returns the value of the last modified date. This escape actually removes all the white spaces and special characters, converts them into escape sequences.
How do you access a Java applet from Javascript code?
This we had seen also. The applet when you are creating in the APPLET tag in the HTML file you can give a NAME attribute to it. For example the name is myapplet. Then you can access that Java applet and Javascript by referring to the object as document.myapplet like this.

(Refer Slide Time: 58:30)



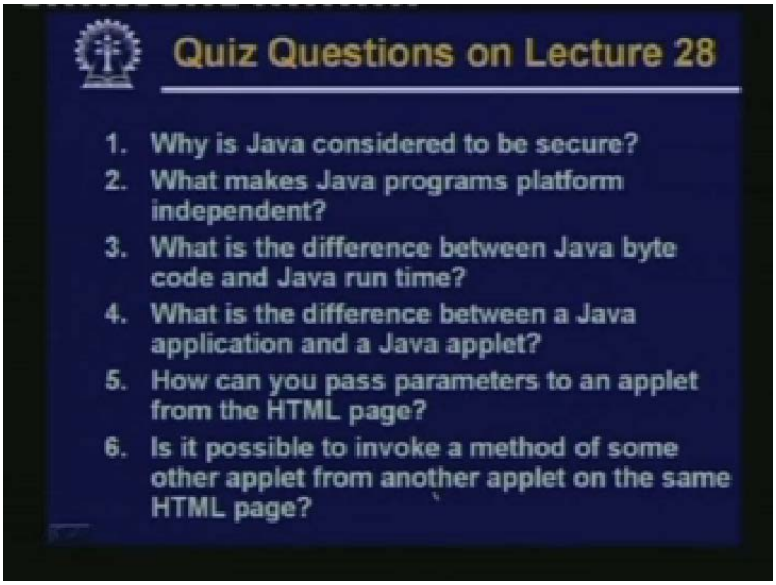
What do the window and document object represent?
This again I have explained repeatedly document object represents HTML document where as window object represents the window or the frame, where this document is being displayed.

(Refer Slide Time: 58:50)



How do you change the background and foreground colors of a document?
By using the bgColor and fgColor attributes you can set them to any desired color. So background and foreground colors can be changed. So some questions from today's class.

(Refer Slide Time: 59:07)



Why is Java considered to be secure?
What makes Java programs platform independent?
What is the difference between Java byte code and Java run time?
What is the difference between a Java application and a Java applet?
How can you pass parameters to an applet from the HTML page?
Is it possible to invoke a method of some other applet from another applet on the same HTML page?
So with this we come to the end of this lecture. Thank you.