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

Lecture – 17
Packages – I

While, we are discussing about information hiding several times we have discussed about the pack, we have mentioned about the Package in Java. Now, package is basically a very important concept in Java. So, in today's module we will discuss about packages that those are possible in Java programs. So, concept of package will be discussed first. Now, what exactly the package it is?

(Refer Slide Time: 00:45)

In fact, there are very two powerful and very innovative features that is contributed by the Java developer, is called the packages and interfaces.

So, that is why Java is so versatile and Java is so popular nowadays because of this concept actually. So, these are the two unique features packages interface. So, today in this module we will learn about the concept of packages, interface will be discussed in other modules.

(Refer Slide Time: 01:19)
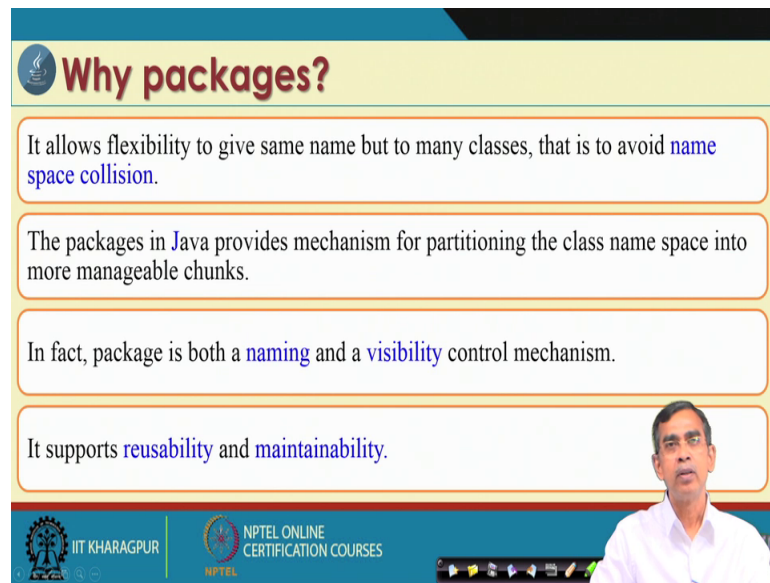
Now, a package is what? So, so far the technical term the package is concerned as you know package, package means it basically a container. So, it is basically container of what? Is a container of classes? So, in other words a package is basically is a collection of is of a set of classes. So, it is basically an idea about the space compartmentalization; that means, we can write we can compartmentalize all the entire space, the programming space into the different packages. The idea it is that because Java can be used to solve many problems and suppose you are a programmer in your organization which basically dealing with only the graphics related program.

The other programmer may be related with other team of developer may be related with networking related. So, all the classes those are related to the graphics they should be compartmentalized into one place that is called the package. On the other hand those are the teams the team members who are working with say networking they should compartmentalize their space by making their own package. So, this concept is called the package there and as we know you can recall Java JDK, the Java developer kits itself includes a large set of packages also. So, these are called API, that Application Programmer Interface or built in Java packages are there. So, they also develop based on this concept.

(Refer Slide Time: 03:03)

Now obviously, why packages? It has many advantages. So, definitely it allows flexibility to give same name, but too many classes that is to avoid namespace collisions. Because, say suppose your package is graphics you can get a name class a, another package who are working with this one they should not have be restricted giving the same name; they can give same name to their. So, basically name you can give the name of your son, same name can be given by other parents to their son like this one. So, namespace collision if you want to avoid it then definitely the concept of package should be allowed.

Now, here also as we already told you that concept of package allow, it is basically a mechanism for partitioning the class name space into a manageable chunks. Because, all related things are to be placed together. Graphics related all programs all classes should be placed together. So, is a better practice and it also help naming and visibility control mechanism, visibility control something is accessibility. So, some if we use a package some accessibility can be automatically imposed. And then this is a last, but not the least is that it supports reusability and maintainability. If a package, if it is there you can create one package of your own considering, the reused of other course those are declared they are in other packages.

So, reusability and maintainability this means that if you want to maintain the previous versions let us keep them in one package, the newer version can be maintained in another

package. So, package concept give you code sharability, code maintainability other than visibility and namespace collision. So, these are the so important process services that can be obtained from the concept of package that is a package is very important one features in Java programming.
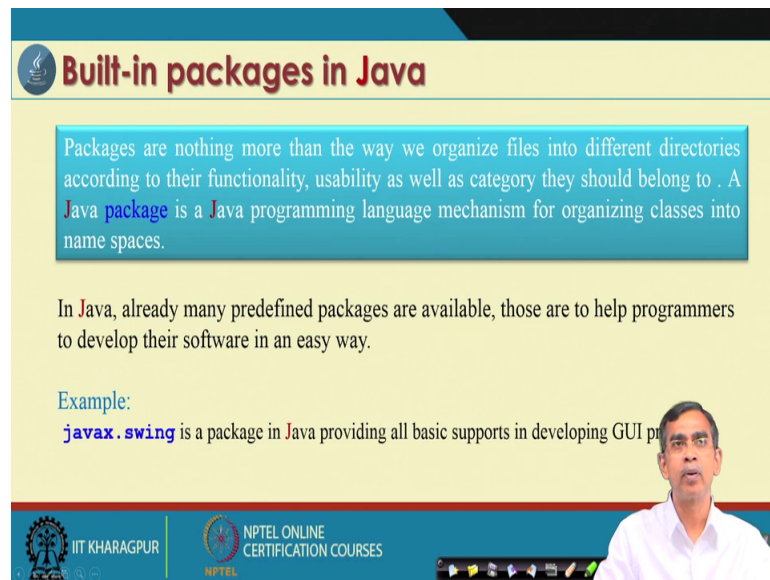
(Refer Slide Time: 05:09)

Now, so package using this package as I already told you the code reusability can be achieved. It helps resolving naming collision, it also help us to control the access specification, information hiding access can be controlled. And, nesting hierarchy a package can be defined within another package. It is in fact, consideration for the maintainability or expense ability otherwise it is there version from one version to another version. So, we can make one sub directory or sub package under a package. So, hierarchical concept of package also can be extended here. So, that more flexibility for a programmer can be obtained. So, these are the several advantage that the package can give to the Java programmer.

(Refer Slide Time: 05:59)
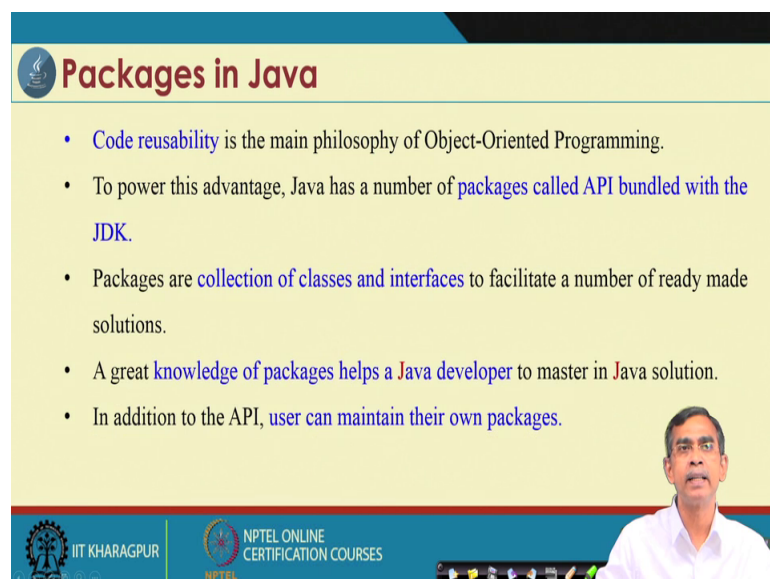
Now, I just now mention that built in package called the API Java. So, there are many built in packages are known in the Java program itself. So, here the built in packages for example, java dot lang is a one built in package as that means, related to input, output, some maths function and everything. Those are automatically pushed into that directory called the java dot lang. Likewise javax dot swing is a very powerful on package for graphical user interface programs programming.

(Refer Slide Time: 06:39)

So, these are the several packages are there that it is there. Now, all these packages basically to ensure the code reusability. That means, you can use the classes those are define in this package in your own class, and you can access it without writing the program of your own. So, it helps a programmer a lot because built in packages are really very powerful boon to the Java programmer, help to the Java programmer.

Now, all the packages those are called built in packages are bundled in API and then there basically JDK. And, whenever you install JDK automatically all these package will be installed in your working directory or bin directory. Bin directory is a executable file the all commands javac, java, java h, java doc all these commands are there. It is stored in the same directories there. Obviously, you have to discuss the setting of class path. We will discuss shortly about this one and then package is basically not only that classes, it also another concept of encapsulation. It is called the interface, it is also collection of classes and interfaces interface will be discussed after the packages cover.

And so, API is the built in packages; now what I want to say is that other than API user can maintain their own packages. So, user defined packages so, built in package versus user defined package. Now, all the user defined package suppose you developed one code, that code you can share with other team members. So, you can maintain one package and then shared it. So, these are user defined package.

(Refer Slide Time: 08:19)
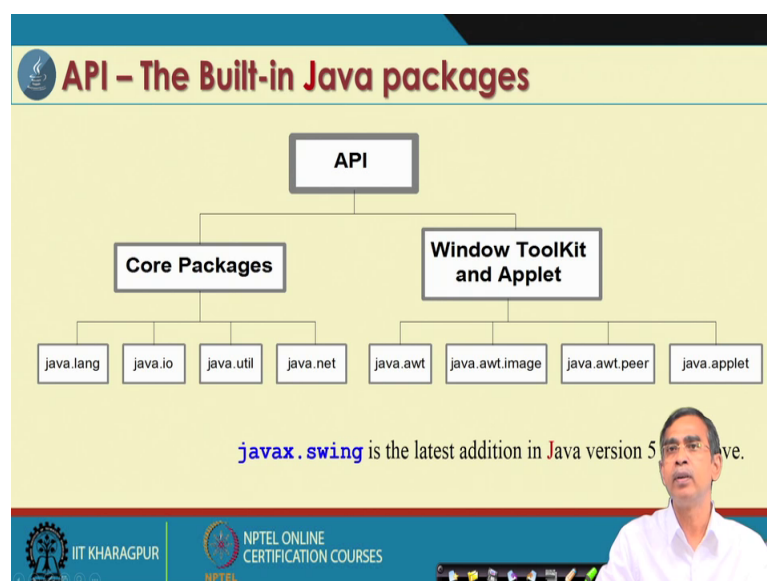
Now, first we can discuss about built in packages and then we will discussed about how user can defined their own package. So, here is the total taxonomy of all the packages those are there in Java programs and as I told you Java can be used for many purpose. So, mainly two purpose the core packages; using this core package one can develop any application using java dot lang, input output, util. These are the different data structures and then networking related different methods those are discussing java dot net package. So, these are for core programming the Java core programming, called Java core packages that can be helps to the Java core programmer

Other than this core programming, the graphical user interface related programming those are basically done by using the package called AWT Abstract Windowing Toolkit in AWT image peer multimedia so, many things. So, many classes are there and then java dot applet is also another very powerful package related to the applet development. And, other than this thing javax swing is also latest inclusion in the windows toolkit and applet packages. So, these are the total the packages, 9 packages I should say are available to the Java programmer. Here one packages also miss, it is called the java dot sql for that is also Java database connectivity anyway. So, these are the so, many packages are they are bundled in API and then available freely whenever you install the JDK.
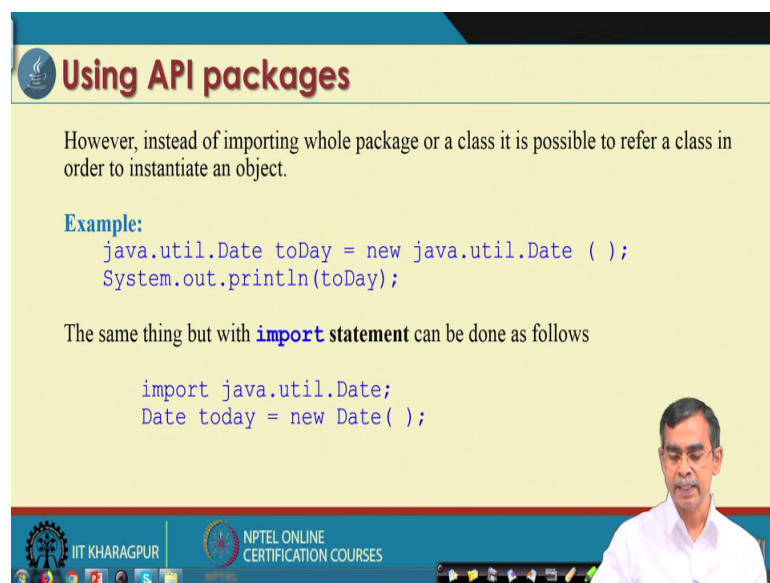
(Refer Slide Time: 09:57)

Now, so a package as I told we can use these packages. So, these packages can be used; that means if that a particular package you want to use in your program and that concept

is basically you have to follow the input command. So, input statement we have used earlier also without knowing the details meaning of that, now you are going to learn about it. So, input is one the statement that can be placed at the very beginning of your class declaration or program.

So, input can access the particular package that for example, input if we write input then java dot lang dot String this means that we one you want to use string class which is declared in java dot lang package in your program. And if you want to access all the classes which are declared belong to a particular package, then instead of that specifically naming the class you can write java dot lang dot star. So, star indicates that all. So, this we basically access the all packages that you want all classes belongs to a particular package you will be able to access it.

(Refer Slide Time: 11:09)

Now, here the other way of around also the package can be accessed. For example here if you see say, suppose you want to access one class Date. Date is basically the built in class which is defined in util package in java JDK. And, here you can see with this statement, what we are doing. We are using a built in class called the Date class and creating an object of that class called toDay. So, here toDay object is created which is a class of type Date, the Date is not by defined by a programmer rather it is a built in class.

So, this is the way that it can be used and then whatever the method it is allowable for this object which is discuss according to this class you have. So, for the programmer on

knowledge that is required is that, if you want to use the built in package you should know what are the classes belongs to this package. And, what are the methods belongs to this package and what are the constructors belongs to that class. So, landing all these things makes a lot for a Java programmer actually, but it is very difficult job because it is a huge. So, many 9 packages are there; in every packages around 20 classes if we take approximately.

And, in every classes there may be around 4-5 average constructor. And, in addition to these there may be around 10 methods and all the methods are having the different arguments overridden method like this one. So, learning but with tedious, but anyway constant involvement in programming and using makes a programmer habituated to the different method. And, it basically gives a lot of skill to the programmer. Anyway it is a matter of practice actually, you should have enough practice then you are the master programmer in Java actually.

Anyway so, this is the idea about that how one can access a class who is belongs to a API and this is also another way right import. And, then using this import we can create an object that is also alternative method by this also. Whatever the way we can access it, if you know that this class belongs to this package that that understanding is important. Once this understanding is done with you then you can use this thing, as if it is your class defined in somewhere in some directory like. So, this is the concept is very simple in that sense actually.

Now, here we will discussed about user defined packages; that means, we know that API packages are there we can use it, learning is required that is all. Now, I will discuss about that how a user can define his or her own packages in its program. So, this is the defining packages in Java program. Now so, as I told you package is a directory, is a directory, subdirectory whatever it is there. Now, you have to create one directory where, you can store all the files. I mean file means the Java file not only the Java file, Java file in addition to all the Java file the they are corresponding class file.

If you store dot java and the corresponding dot class then you can put all those things into a directory and then that directory become a package. The things appears very simple is not it, but actually there are many more things to be considered. It is not so simple, but in a simple word I can say like this, if we want to create one package of your own. So, create a subdirectory first. Now then to mention, that this is your package then before going to declare the class. So, you should use one keyword called the package. So, here package is the keyword and MyPackage is the directory where, you want to keep all your class file and Java files into that.
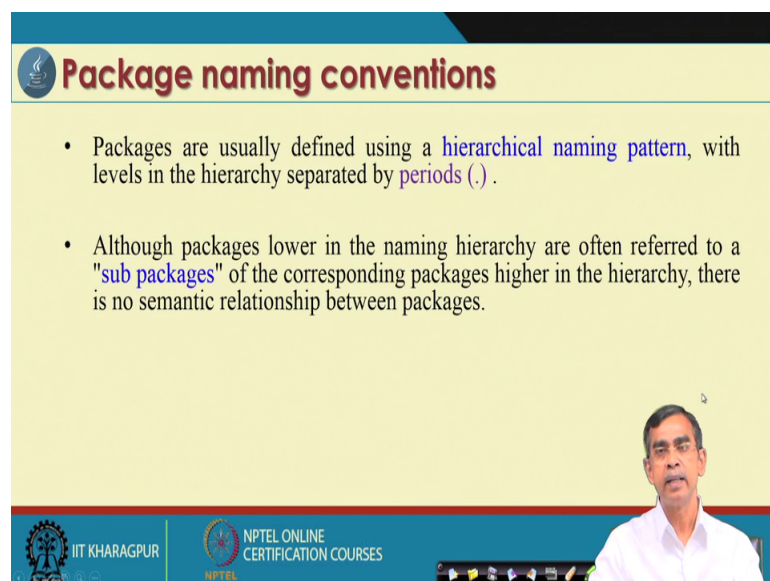
Now, this statement is basically tell that this is the class that you have built it, is basically belongs to the package MyPackage. So, if you is these kind of common for all class that you have created and giving the corresponding package name, then automatically all the packages will be created. So, a package can contains any number of class file, any

number of Java file there is no limit. Absolutely theoretically there is no limit, you can go on putting whatever it is there, but one thing is that it is obvious two classes having the same name not allowed.

Because, it will not allow overwritten latter one will overwrite a previous one. So, you have to very careful whenever you do it and is in fact, it will not help you. Because, after once you create one a dot java next time whenever you create a dot java definitely at the time of saving the system will ask that ok; do you want to overwrite. Then you overwriting means the previous Java file. So, it is basically the in the same package the two files having the same name is not allowed. However, in two different package the same name can be allowed two different classes.

So, this concept is that that how user can create a package. This is a packaged what is the statement I can say, the package statement is used for declaring one package and then this class then will put into these packages. So, concept is like this.

(Refer Slide Time: 17:05)

Now, so this is the idea about it and hierarchically also package naming can be there using by dot core; you know directory under the directory subdirectory, under the sub directories sub directories all these things also quite possible. And, if you want to access one particular class into particular sub directory so, dot periods is sign is called. So, basically the concept as it is there is an operating system concept x dot y dot z that mean x is the root.

Then under x y is a directory under y z is the sub directory and in this z there may be some file say dot a. So, x dot y dot z dot a specify which is a basically location of a particular class belong to that directory like. So, this is the periods that can be used to have the resolution of the; or we can say that location resolution that ok; we can specify which is the location of a particular class or particular package. So, this is the concept of hierarchical definition is basically concept of sub packages in that sense.

(Refer Slide Time: 18:13)

Now, so here again so far the naming of the packages is concerned there is no rule of course, but the rule is that the way you define directory you should do it, but that space is not allowed. So, whenever you declare a sub package usually all small capital small letters case right, small lowercase characters should be used better to avoid the uppercase. But, sometimes the past is small I am upper a lowercase letter and then maybe say MyPackage m. So, MyPackage we can write.

So, MyPackage so, my and then pa P is the capital letters and this is also and that I can write as the package. So, it is like that that kind of con set it is there. So, this is also you can declare about it anyway. So, the naming is very for example, all the API you see the name is I think 3 to 4 character dot net, dot lang, then dot io this kind of things and names should be small enough but, it should be unique of course. So, this is the concept that so far the naming is concerned.
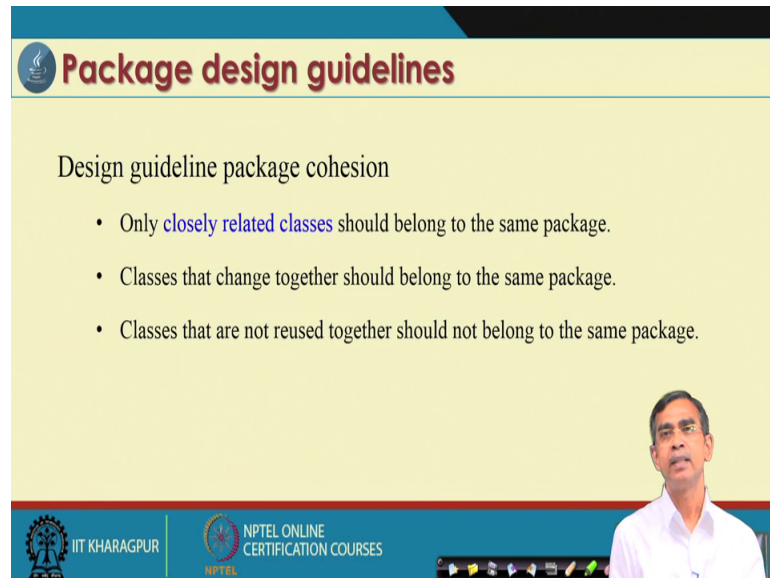
(Refer Slide Time: 19:29)

Now, we will discussed about in addition to these things few things are very important to remember; I just mention this. This slide is very important in that sense, now user can define their own packages, and if a user define a package then that package statement should be first statement in the class file creation. So for example, if you want to create a dot java a class and it is belongs to a package say my package. So, first statement should be packaged MyPackage and then public class a dot java whatever you can write it like that.

Now, so if you define a class which belongs to who should belongs to a package then that class should be declared as a public. So, it should be declared at a public because, you should give the access to other classes rather other programmers to access this one otherwise there is no concept of packages coming. So, all class file belongs to a package should be declared as a public. If you declare private you cannot use it protected cannot be declare any class cannot be declared as a protected class of course, and you should create a subdirectory under the working directory; this is very important with the name same as the package name that you have mentioned.

So, working directory where your program will run all the packages should be under that working directory that is important. Now, regarding this thing you can set the class path environment variable in the setting of the system or your path setting also you can do, regarding this thing we have already discussed very beginning. And another important

thing is that you create a file containing a class that is the dot java and that file should be compiled successfully. That means, dot java should be stored along with its compiled version; compiled version is bytecode. So, all the dot java as well as the dot bytecode should be there. So, these are the essential condition so far creating the package is concerned. So, these are the 5 points agenda is very needs to be adhere to very sincerely.

(Refer Slide Time: 21:41)
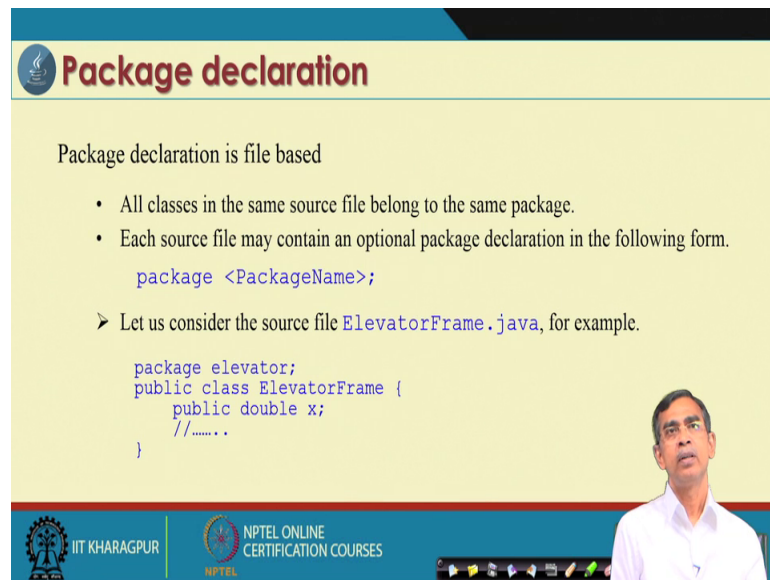
Now, another design guideline is that the package should be organized very efficiently. So, that all related class related to a particular project, related to a particular application development, related to a particular utility all these things should be placed into on package. It is not that a very large package all the package does not have any limit can be put there it is not a good practice although it will work it, but so far core maintainability, reusability is concerned this is not the good practice. So, this is the one thing that you should follow it is here.

(Refer Slide Time: 22:15)
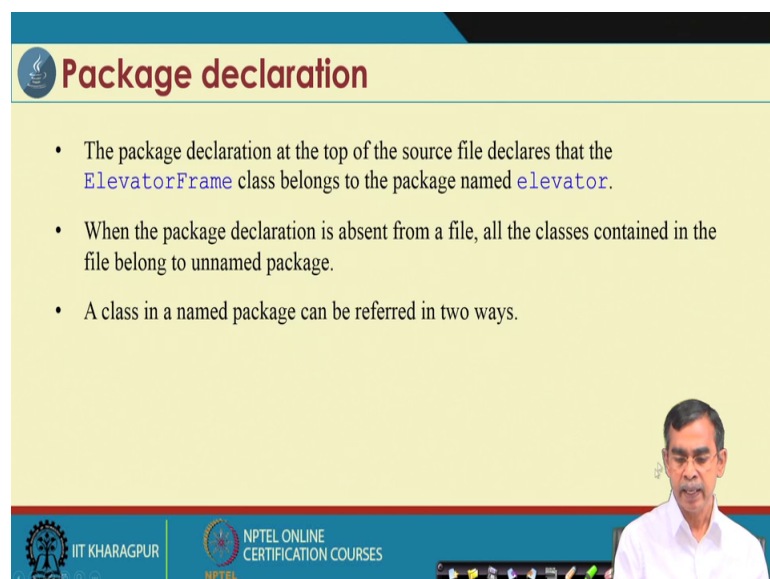
Now, again repeating the same thing, you can declare a package by breaking the package name. It is basically the directory working directory under working directory we can say here is an example. If this is the one your dot java file you can put it into a package, let the name of the package be elevator and then this is the declaration. And, as you see it should be declared as a public. Those are the things as I have already discussed, I am just repeating because it is very important right concepts.
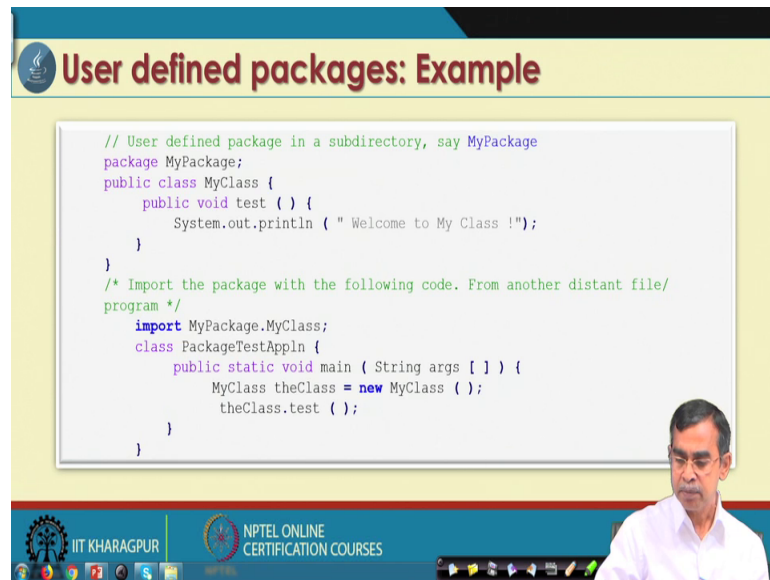
(Refer Slide Time: 22:47)

So, packages declaration follows this kind of systematic procedure. We have discussed about and naming can be done in that way as we have already discussed about. Now a class in a named package can be referred in two different ways that we will discuss shortly after discussion of few more important things here.

(Refer Slide Time: 23:13)

Now, here is an example let us consider. This example is basically is an attempt, this is this example is an attempt to define packages. The name of the package is MyPackage, as I told you two different way that you can define it I am telling you fast way. So, this is the one I have declared in one package here one class in a package. The name of the package is MyPackage and name of the class is My, I can use this way it will work for you actually no problem. So, here the subdirectory MyPackage is created and in the subdirectory all dot java and class file is stored there.

Now, another way here one this package is created and then the same package can be used in other class. Here is that if you consider this is the other class where you want to use it; so the import statement is there. So, import then if you want to access a particular class so, MyClass. So, import MyPackage dot MyClass that mean this basically gives you as if this pack this class is in the same file where this program is there. So, this is the idea about that how the package can be utilized.

So, there is two concepts so far the package is concerned; package to build a package and import to access a package in your own program. So, this is the concept it is there. Now,

so far this accessing is concerned there are a few more things are to be learned about; all these things we will discussed in details in our next module. We have planned our next module also to cover packages anyway.

(Refer Slide Time: 25:03)
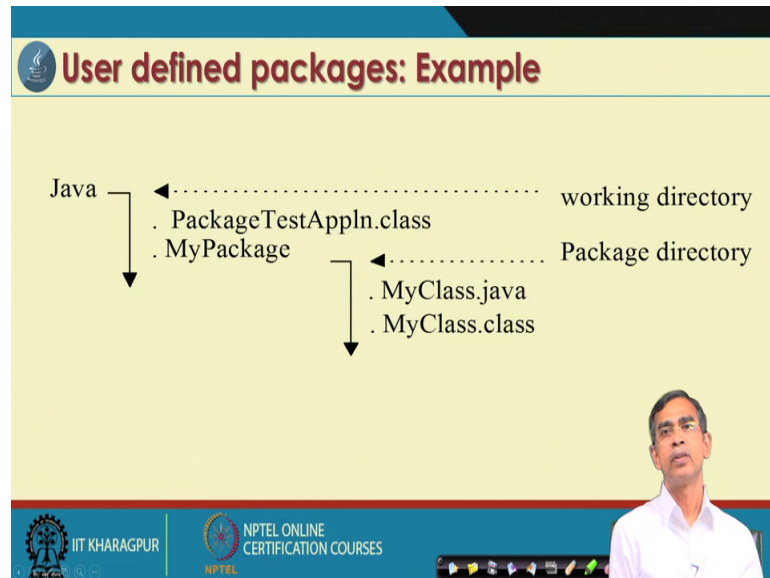
So, as a summary again so, in a Java whatever your bin directory; that means, all Java executables are there you can keep it there. And, if you want to plan your package then you if this is your working directory under this working directory you can create your MyPackage and all these Java class file belong to this package can be put here. And, and as a parent directory of this one just hierarchical up right; so, this class file where your program will include. So, working directory so, this is basically working directory, this is your package directory and all these things will be the bin directory is there.

So, this is the common concept of hierarchical way the package it is there because, you are going to learn many topics in this course. So, you have to maintain many programs those are under demonstration and illustration in the slides and everything. So, my suggestion is that you make a very nice organization of all the things may be say inheritance, one directory working directory. Related to inheritance all the classes that you can you can put into the package and then used your program, use the input command; all the package can be accessible to this either import the package name dot star or import package name dot particular class you can use it.

So, this is the basic concept that you should follow. For a good programmer should a systematic a meticulous method needs to be followed.

(Refer Slide Time: 26:31)

Now, user can define more than one class belongs to a package and it is also not good practice is that all the class should be declaring one dot java file because, you have to compile it a individually and separately. So, basic idea is that all class file should be created in a separate manner. So, here is an example for example, so here class A if it is a component in a package P. So, you can create this one file as it is.

So, A dot java and A dot class and then the next another class see suppose B you want to create so B you can create as a another right. So, this is basically another class we can say this is another class we can say that is belong to the same package P. So, class A and class B; now two classes are created and then they are placed into one package called the P. So, P is the package here in this case.

(Refer Slide Time: 27:45)
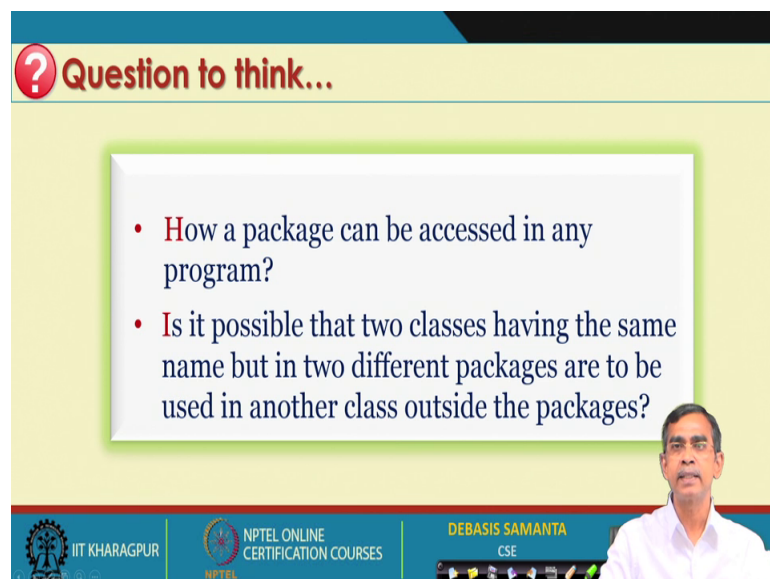
So, it is like this so, more than one class multiple classes can be mentioned in the same file. So, these are the simple step that you should procedure. First you have to create a directory called P and all the classes should be saved dot java with their name and then they should compile the class file. And, both all this dot java dot class file should be stored there. And finally, if you want to use any one class belongs to a particular package use import. So, this is the usual procedure that is there so, for the class package maintenance is concerned.

(Refer Slide Time: 28:21)

Now, we have discussed about how you can defined declare your own package in Java. Now next obviously, question that it is there; if packages are there how I can control the different access specification in it. We have we have an idea about how the information can be controlled by using for access modifier in our one module discussing this course. Now, we will discuss about that modification those are applicable two packages also.

Now, so this thing will be discussed in our next module. Our next module will cover package again, but package mainly we will discussed about the naming convention, the setting of the different variables so, far the system variables, environment variable is concerned and finally, the access modification. So, this we will discuss in our next module ok.

Thanks for your attention.