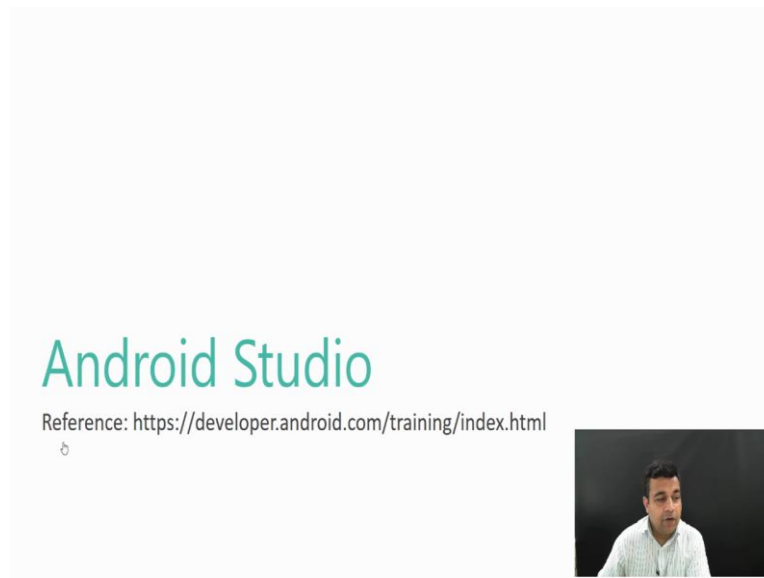


**Mobile Computing**  
**Professor Pushendra Singh**  
**Indraprastha Institute of Information Technology Delhi**  
**Lecture 10**  
**Android Studio**

Last week gave you a quick introduction to android program. You develop a simple hello world application and another applications slightly more and good. And then you deployed it on an emulator and also deployed it on a phone. Hopefully all of you have done that. In this week we start with the revision. But while doing a revision we will go into the detail. So let's start with your android studio one more time.

(Refer Slide Time: 0:48)



For all these videos I will be using the material from [developer.android.com](https://developer.android.com) website. This is the official android developer website support by Google. And has all the required training material that you need to learn android programming.

(Refer Slide Time: 1:06)



## Setting up Android Development Environment

- System Requirements
  - Windows 7 and above or
  - Linux system with version 2.11 or above of GNU C Library or
  - Mac OS X 10.8.5 or later
  - Minimum 2GB of RAM (8GB preferred)
  - Approximately 4GB of Hard disk space
  - 1280 x 800 minimum screen resolution
- Latest JDK installed
  - Check your system with command `java -version`
  - Current latest version is 8
- Download latest version of Android Studio from
  - <https://developer.android.com/studio/index.html>

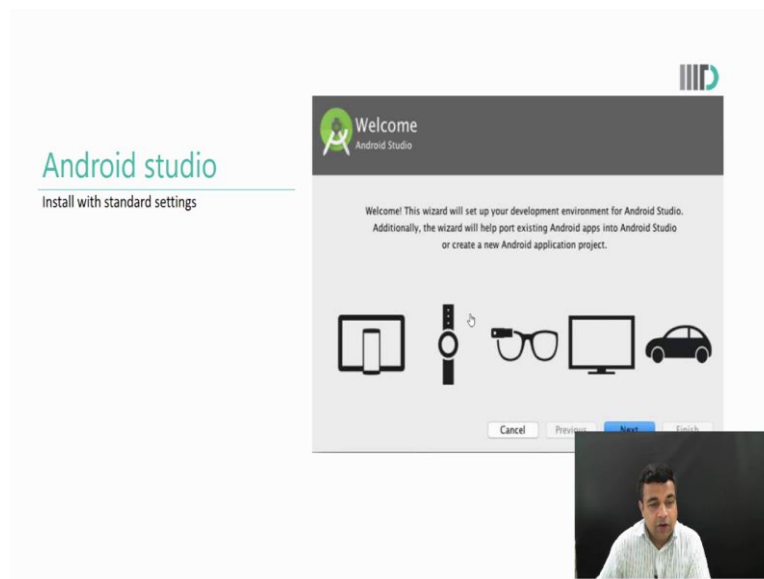


Okay some of you have already installed android development environment called android studio those who have not please see the details again. For this you will require have. For this you will be required to have a PC or a Linux or a mac operating system with at least 2 GB of RAM and at least 4 GB of hard disk space available. The more you Ram you have better it will be. Also the operating system must be windows 7 or above or mac OS X10.8.5 or above or Linux system which has a GNU C library version 2.11 or above.

Before installing android studio make sure that you have installed Java successfully. One way of checking it is by running the command `java-version` when you run this command you will see what version of SDK you have installed on your machine. Please note that I use JDK or SDK (())(2:18) they both mean the same.

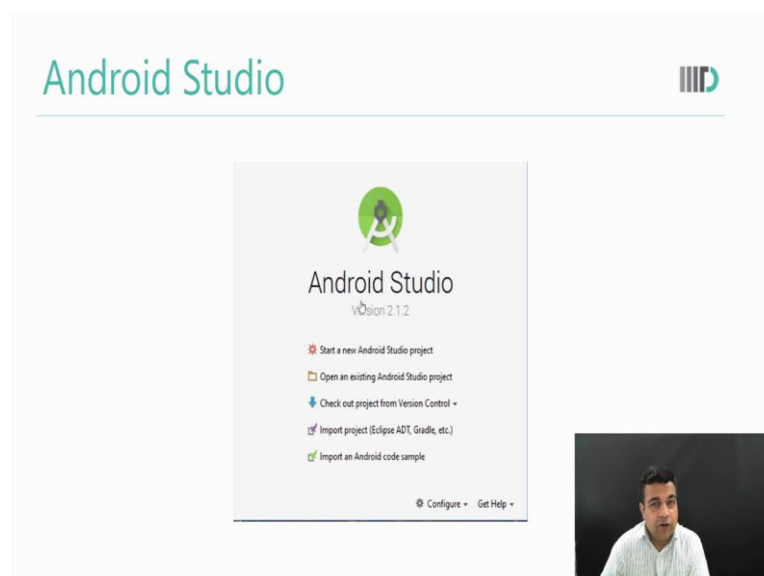
Current latest version of SDK is 8 ideally I have to run this command we should see that your installed version is indeed 8. If it is not then you may want to upgrade your java. Once you have installed java you should download the latest version of android studio from the developer website. The link is given here.

(Refer Slide Time: 2:46)



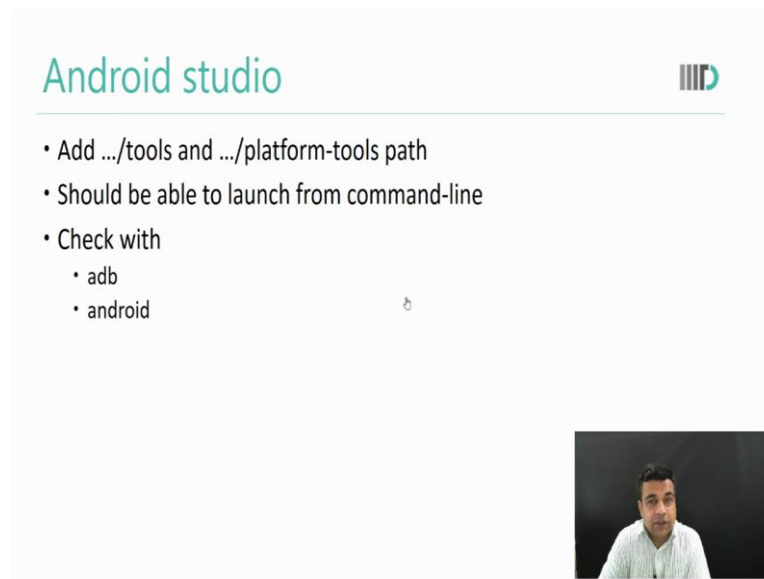
When you start installing android studio it displays you screens all of these screens have a next button. If you go by default setting in the beginning that will be perfectly fine. And you would have got your android studio installed. If you go by default setting that would be perfectly fine and will have your android studio in no time.

(Refer Slide Time: 3:11)




After you have successfully installed android studio you may start it by double clicking and when the android starts it shows you a screen similar to this. It also gives you the version no of android studio that is currently installed. As you can see I have version 2.1.2. As you can see I have version 2.1.2 installed on my machine. If this is not the case you may going to configure and update your android studio.

(Refer Slide Time: 3:46)



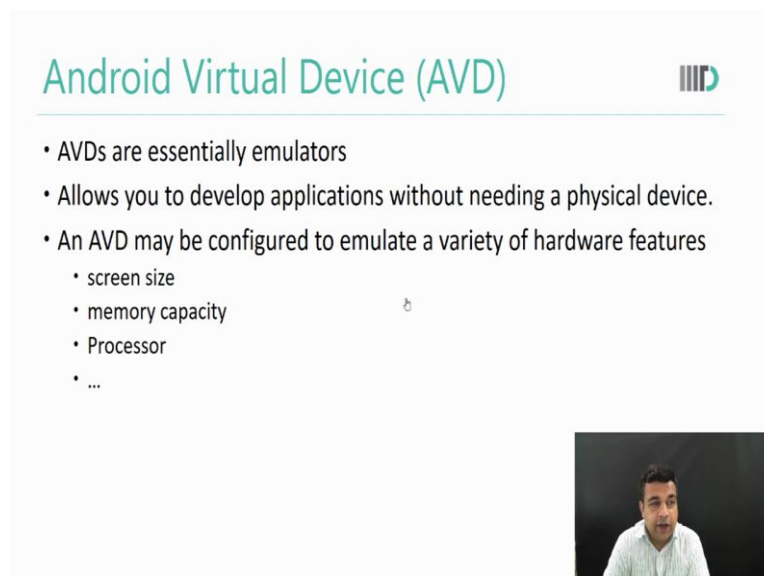
**Android studio**

- Add .../tools and .../platform-tools path
- Should be able to launch from command-line
- Check with
  - adb
  - android




Once you have successfully installed android studio it is good idea to add slash tools and slash platform tools to path of your system. This will allow you to launch these applications from command line. For example once I have given these paths in windows in the environment variable I can just write android on the command prompt and android studio will launch from it.

(Refer Slide Time: 4:12)



**Android Virtual Device (AVD)**

- AVDs are essentially emulators
- Allows you to develop applications without needing a physical device.
- An AVD may be configured to emulate a variety of hardware features
  - screen size
  - memory capacity
  - Processor
  - ...

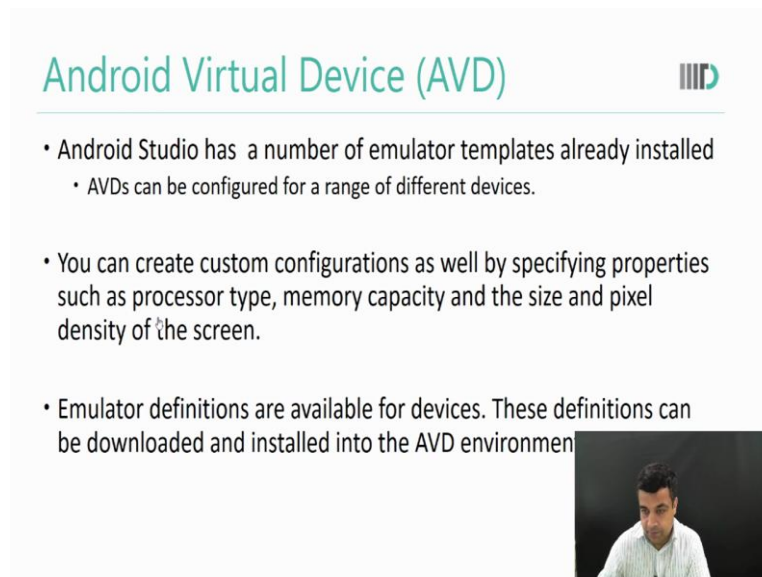


Android studio comes with what we called AVD or android virtual devices android virtual devices are essentially emulators. We use one of those emulators in last week to deploy our applications. Emulators allow us to develop applications even without having a physical device. This is a very use full feature because I may not own the device that I want my

application to work upon. And in fact it is pretty much impossible for somebody to own all the android devices.

However it is not impossible to create emulators with respect to the every device that is available in the market. You may also configure an AVD by specifying different hardware features such as screen size, memory capacity or processor.

(Refer Slide Time: 5:03)



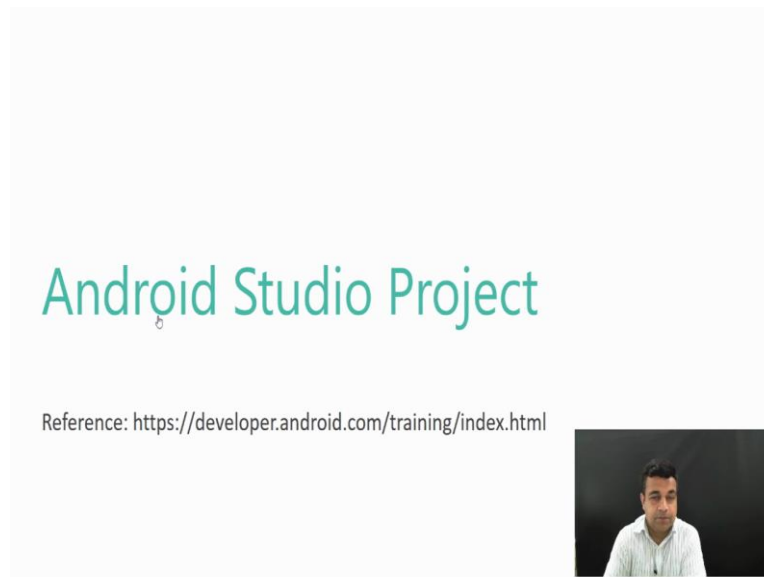
The slide is titled "Android Virtual Device (AVD)" in green text at the top left. To the right of the title is the Android logo. Below the title, there are three bullet points:

- Android Studio has a number of emulator templates already installed
  - AVDs can be configured for a range of different devices.
- You can create custom configurations as well by specifying properties such as processor type, memory capacity and the size and pixel density of the screen.
- Emulator definitions are available for devices. These definitions can be downloaded and installed into the AVD environment.

In the bottom right corner of the slide, there is a small video inset showing a man in a white shirt speaking.


Android studio has a number of emulator templates already installed and then we can also configure AVD for eventual different devices by creating custom configuration like using all different processor types, memory capacity and size and pixel density of the screen. Emulator definition is also available for devices. These definitions can be downloaded and installed into AVD environment.

(Refer Slide Time: 5:31)



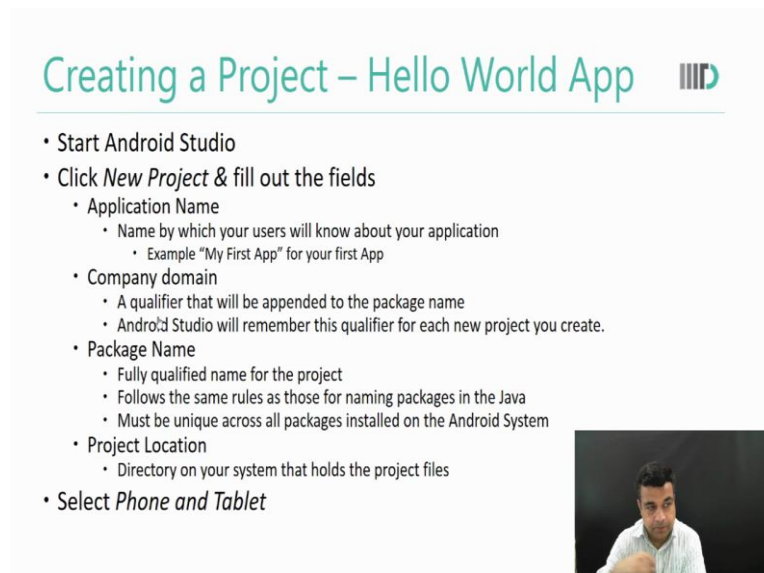
Android Studio Project


Reference: <https://developer.android.com/training/index.html>




Now hopefully by this time you have an android studio installed and you are ready to develop your first project if you have not already done that. If you have already done that in week 1 then I advise you to watch this video and go back to your week 1 project and start running your android studio side by side. And then see what I am explaining here.

(Refer Slide Time: 6:00)



Creating a Project – Hello World App 

- Start Android Studio
- Click *New Project* & fill out the fields
  - Application Name
    - Name by which your users will know about your application
    - Example "My First App" for your first App
  - Company domain
    - A qualifier that will be appended to the package name
    - Android Studio will remember this qualifier for each new project you create.
  - Package Name
    - Fully qualified name for the project
    - Follows the same rules as those for naming packages in the Java
    - Must be unique across all packages installed on the Android System
  - Project Location
    - Directory on your system that holds the project files
- Select *Phone and Tablet*



You start creating a project by clicking new project it displays you four fills. The first fill is application name. Application name is nothing but the name by which your application will be known to end user. For example when Facebook develop its application it named it Facebook when Indian railway developed its application to reserve railway tickets it named it

IRCTC similarly you may name your application anything that you want. Let us say my first app or math quiz which we did last time.

The second fill is of company domain. And the third fill is of package name. I would like to explain both them together. Package name has the same concept in android as it has in java. The package name specify where your class exist among other so class. The package name is specifies various class exist among other classes of java that may be available. So on the play store your package name must be unique which will mean that your application classes will be unique.

The next fill is company domain and after that this package name. I will explain both them together. Package name follows the same rule as it follows in normal java. In fact package name is specifies your class name in unique manner so must have a unique package name for all your classes now how do you make sure that you have a unique package name for all your classes. Well one way of doing this is to prefix it with your company domain name. Because you know that your company domain name is unique you can be sure that your package name is also unique.

And that's why once you give a company domain name into your android studio android studio remembers it and uses it as a qualifier for each new project that you create. The last fill is project location which is nothing but a direct location in which you will store your project. Once you have filled all these fills and click next. You come to a screen which asks you for the target currently have a target is phone or tablet. Android studio also gives us choice to develop partition for android wear and android class and android TV. In this course we will be only developing application for phone and tablets so we choose phone and tablet.

(Refer Slide Time: 8:55)

## Choosing Minimum SDK

- The Minimum Required SDK is the earliest version of Android that your app supports, indicated using the [API level](#).
- Setting a lower version allows your application to be deployed on higher number of devices
  - To support as many devices as possible, you should set this to the lowest version available that allows your app to provide its core feature set.
  - Android Studio provides a hint on % of devices that can support your app when you choose the minimum SDK
- Setting a higher version allows you to use latest features while limiting the deployment.
  - If any feature of your app is possible only on newer versions and it's not critical to the app's core feature set, you can enable it when running on the versions that support it

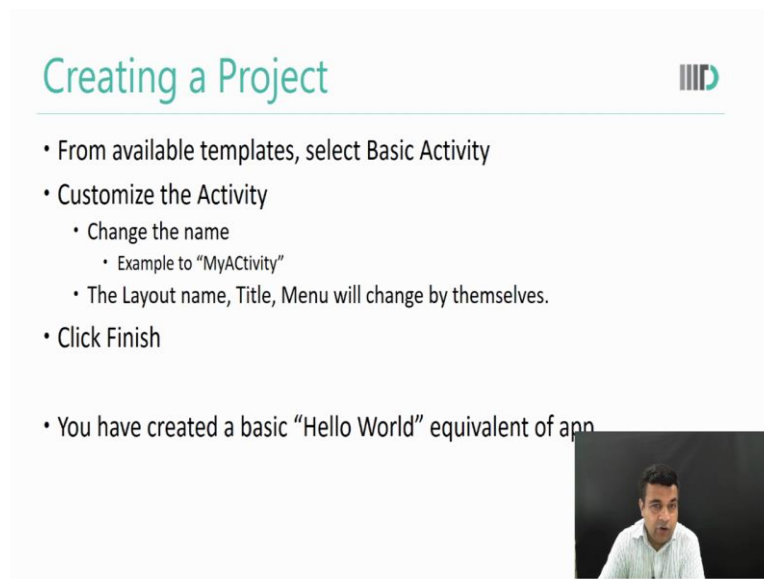


The next is also very important the next screen ask you choose from a minimum SDK. A minimum SDK is the earliest version of android that your app supports. And this SDK level is indicated using the API level. For example you may have API 23 or API 21 or API 24. Setting a lower version allows your application to be deployed on a last number of devices. Instead in fact in fact when you choose a version number android gives you a hint about the number or devices on which your app can be deployed.

You may try to choose different API levels just to check how the percentage of devices change. The lower you go higher percentage you will cover. However the lower versions you choose you will however if you choose a lower version then you will not be able to use the features which are available in high levels. So if you want to use the latest feature choose a higher version if you want to choose if you want to deploy your application on multiple devices if you want to deploy your application on a lot many number of devices choose a lower number. This a trade of which you decide depending on your application and target audience.



(Refer Slide Time: 10:33)



The slide is titled "Creating a Project" in a teal font at the top left. In the top right corner, there is a logo consisting of three vertical bars of increasing height followed by a teal circle. Below the title, there is a list of steps:

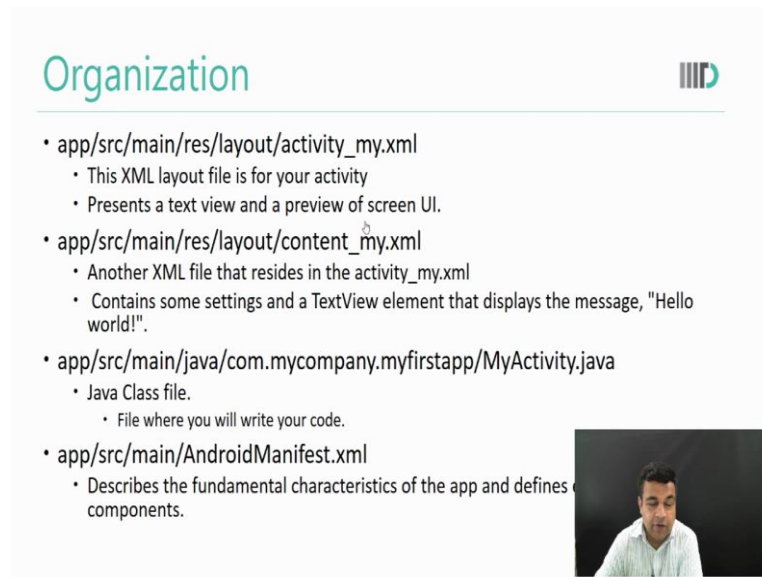
- From available templates, select Basic Activity
- Customize the Activity
  - Change the name
    - Example to "MyActivity"
  - The Layout name, Title, Menu will change by themselves.
- Click Finish
- You have created a basic "Hello World" equivalent of app

In the bottom right corner of the slide, there is a small video inset showing a man with dark hair and a light-colored shirt speaking.

The next is activity templates android provides you different activity template such as basic activity, empty activity and various other activity. In the beginning it is good to choose just a basic activity or even the empty activity. And then android ask you to give your activity name. You can give your activity any name just make sure that your first letter is cap capital. Just make sure that your first letter is capital and activity is the part of the name.

This is an android convention. Once you keep the activity name android studio automatically changes the name of the layout, title and menu. Let them be as it is. And you click finish and (( ))(11:27). You have created your hello world equivalent of app. In fact this app will display you hello world though you need not though so file need not wait in hello world anywhere. I will explain that to you very quickly.

(Refer Slide Time: 11:42)



## Organization

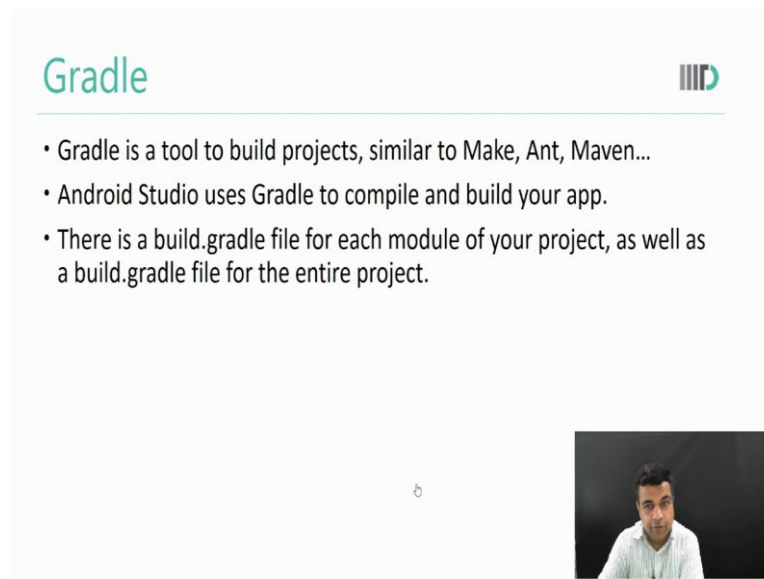
- `app/src/main/res/layout/activity_my.xml`
  - This XML layout file is for your activity
  - Presents a text view and a preview of screen UI.
- `app/src/main/res/layout/content_my.xml`
  - Another XML file that resides in the `activity_my.xml`
  - Contains some settings and a `TextView` element that displays the message, "Hello world!".
- `app/src/main/java/com.mycompany.myfirstapp/MyActivity.java`
  - Java Class file.
    - File where you will write your code.
- `app/src/main/AndroidManifest.xml`
  - Describes the fundamental characteristics of the app and defines its components.

First let's go into the organization of your application. We did not cover this part last time however you will be covering it this time. As you go through the organization structures you will find out different folders. And let's say we choose a folder `app src main res layout` in that we have XML file which will correspond to our activity that we have create. So if I had given the name `my activity my XML` will be named as `activity underscore my`.

This is the XML layout file for my activity and this XML layout file provides me a text view and preview of screen UI as you can see. The second file related to my layout is `content underscore my.XML`. This file again content some settings and this file also content the text view element that is currently displaying you hello world. This text view element is this text view element is available directly from android whenever you choose basic activity and that when you see hello world message even it you have not write it.

The third important file is the java. Java file is the actual file where you will write your code this file would again the available into `app src main java slash your package name slash whatever you indicate your activity.java`. And the fourth is the manifest file. The manifest file describes the fundamental characteristics of the app and defines each of its components. And we will go into detail of manifest file in some later chapters.


(Refer Slide Time: 13:33)



## Gradle

- Gradle is a tool to build projects, similar to Make, Ant, Maven...
- Android Studio uses Gradle to compile and build your app.
- There is a build.gradle file for each module of your project, as well as a build.gradle file for the entire project.

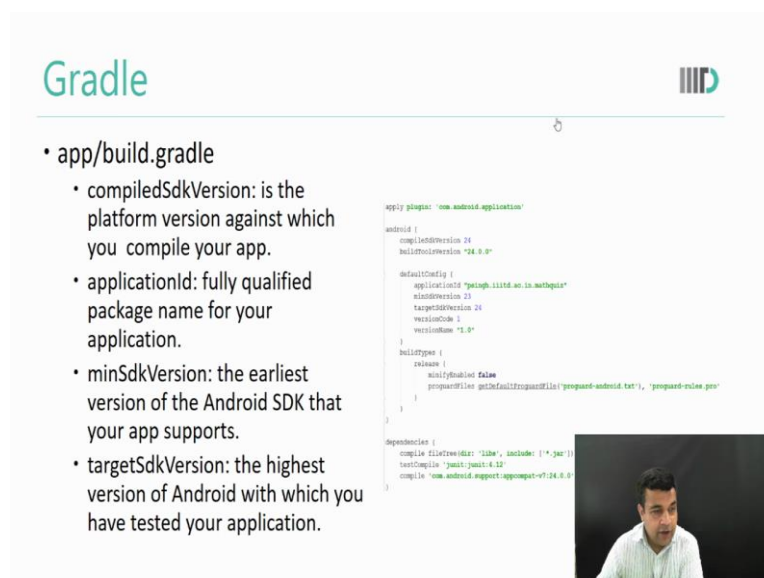
IIIID



Last week you saw that whenever we want to build our application and run it we use to press an icon and it is use to build our application and make it ready for running. This is done using a build tool called gradle. Android uses gradle to build android projects. Android studio uses gradle to build android projects.

Gradle is a tool to build projects which is similar to any other build tool such as make, ant, maven. However gradle is the latest of the build tools (())(14:08) functionality and that why it is chosen in android studio. There is a build.gradle file for each module of your project as well as the build.gradle file for the entire project.

(Refer Slide Time: 14:25)



## Gradle

- app/build.gradle
  - `compiledSdkVersion`: is the platform version against which you compile your app.
  - `applicationId`: fully qualified package name for your application.
  - `minSdkVersion`: the earliest version of the Android SDK that your app supports.
  - `targetSdkVersion`: the highest version of Android with which you have tested your application.


```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 24
    buildToolsVersion "24.0.0"

    defaultConfig {
        applicationId "org.gradle.sample"
        minSdkVersion 23
        targetSdkVersion 24
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:24.0.0'
}
```

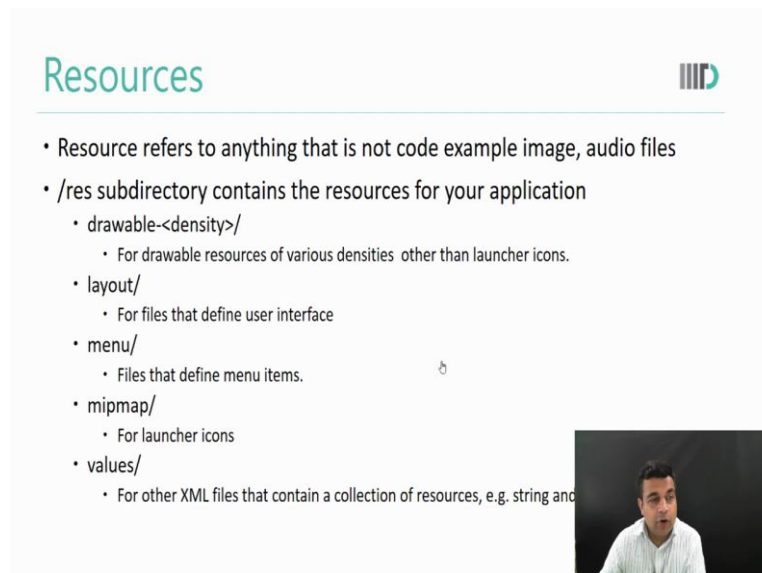
IIIID



As you see I am displaying my build.gradle file and let's see what fills are available. So I have fill called compile SDK version which is named 24. Compile SDK version is the platform version list which I have compiled my application. So because I compile my application for 24 and gradle is 24 numbers. Then there is application ID. Application ID is my fully qualified package name from my application.

As you can see that it has reverse my domain name converted into my package name. Psingh.iitd.ac.in.mathquiz is my application ID then there is a minsdk version which is what you choose earlier. And there is target sdk version which again what you choose earlier. So min sdk version is the earlier version of android sdk that your app supports. While target sdk version is the version of android with which you have tasted your application.

(Refer Slide Time: 15:39)



The slide, titled "Resources", lists the following resource directories and their contents:

- Resource refers to anything that is not code example image, audio files
- /res subdirectory contains the resources for your application
  - drawable-<density>/
    - For drawable resources of various densities other than launcher icons.
  - layout/
    - For files that define user interface
  - menu/
    - Files that define menu items.
  - mipmap/
    - For launcher icons
  - values/
    - For other XML files that contain a collection of resources, e.g. string and

A small video inset in the bottom right corner shows a man speaking.

The next important part in the android project organisation is the resources. Resources refer to everything that is not code. So if you are using an image that is a resource if you are using an audio file that is a resource. In fact even strings are called resources in java and android. In fact even strings are created as resources in android. So here you will see slash res some directory which contains the resources for your application. Inside this sub directory you will see another directory called drawable.

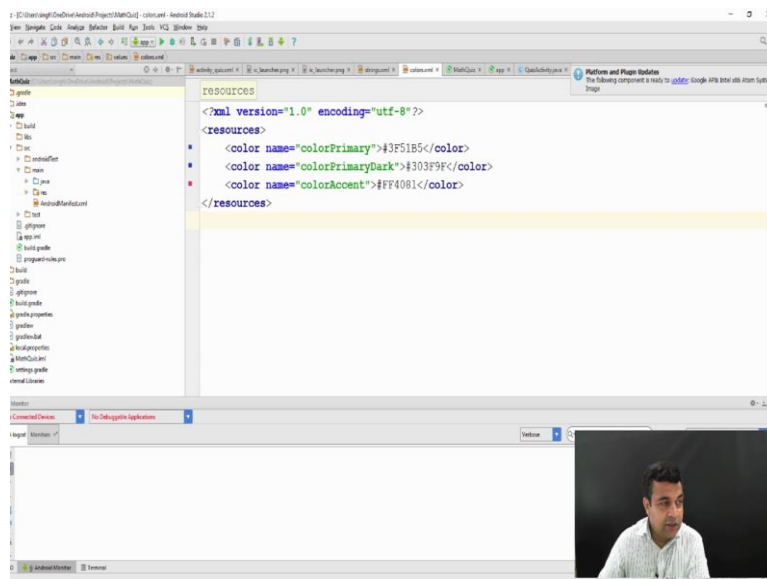
Drawable contains all the drawable resources of different (16:18). The drawable contains all the drawable resources of different densities. But not the launcher icons. Then another directory is layout which contains file that define user interface you already saw activity

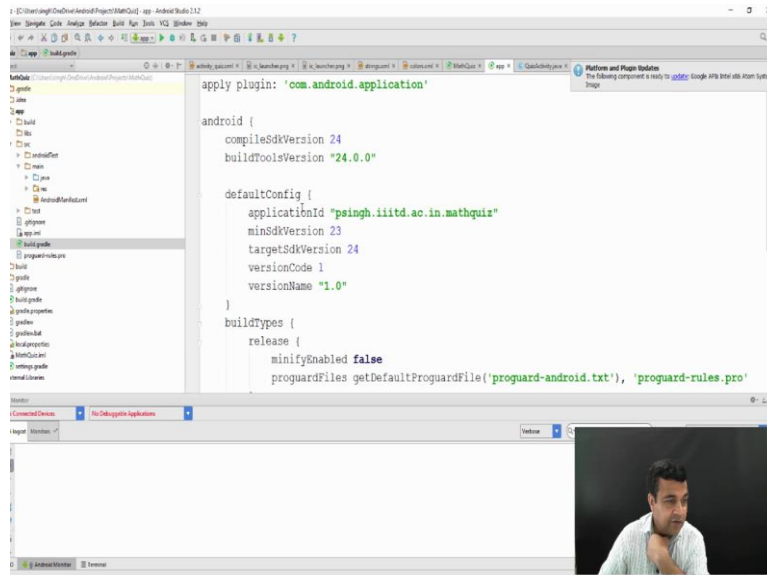
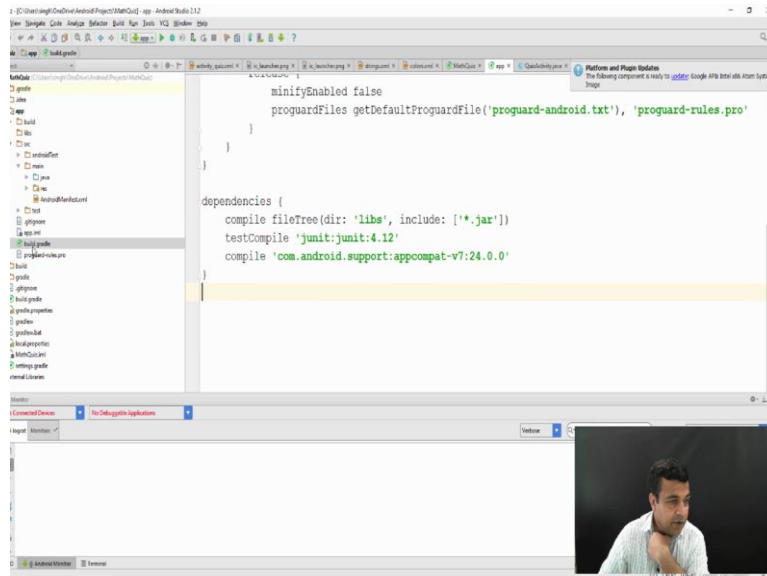
underscore my dot XML as part of the layout folder. Then we have menu folder where there are files that define menu items. Then we have mipmap which contains our launcher icons.

Then we have values which contains other XML files that contain a collection of resources such as string and colour dimensions. This is all about android studio now let us see in our live project. Let me run my android studio and is great to use the same directory structure that we are talking about. You may cross check it on your machine by running your android studio. And while my android studio is starting please go to developer.android.com website and try to read the (())(17:36) material that is given there.

This is your best resource some of you have also been asking for an android text book on(())(17:43). I have already given a name of few text books on the (())(17:48) . a text book which I am using heavily is called as beginner (())(17:52) guide you type on Google you can go on the website of this book. This book is also available on amazon.com. Okay so my android studio have started let us see.

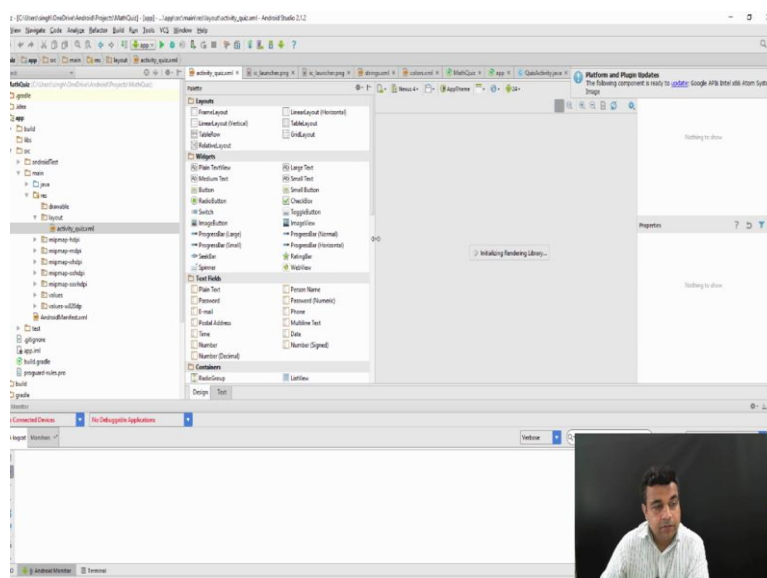
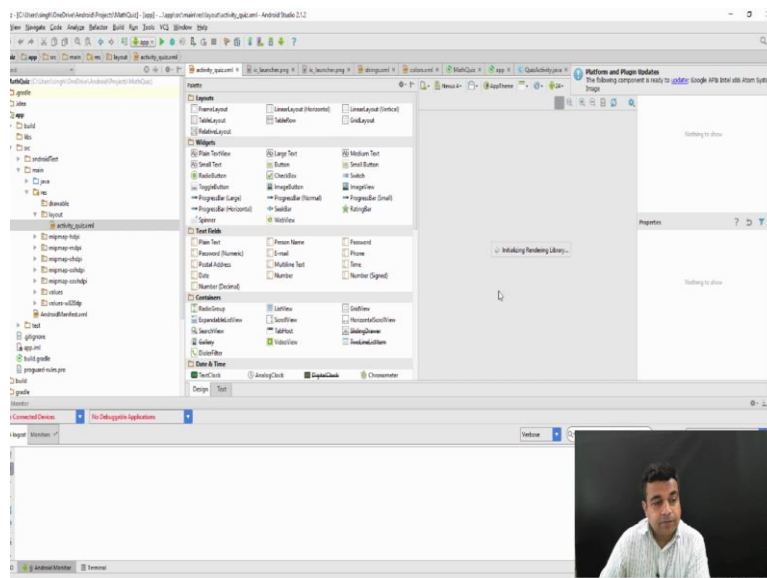
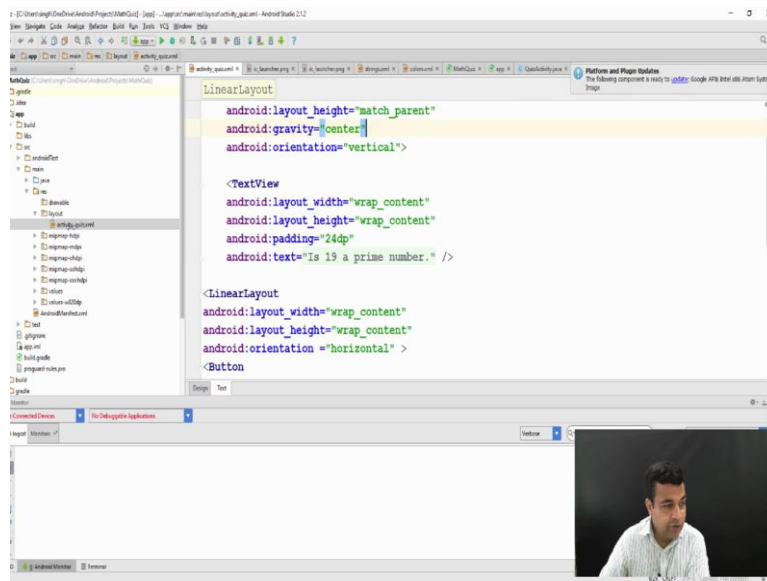
(Refer Slide Time: 18:07)

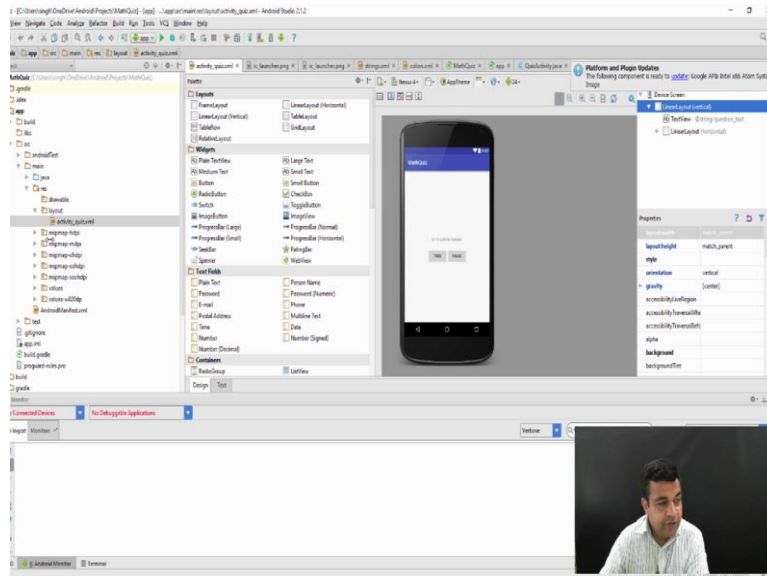




So this is the project math quiz which we have developed lets go into the app and as we can see first lets see our build.gradle as you see we can see the compiler version application ID minsdkversion, target sdk version, etc . Then go to our resource directory we can see the drawable. We can see the folder layout.

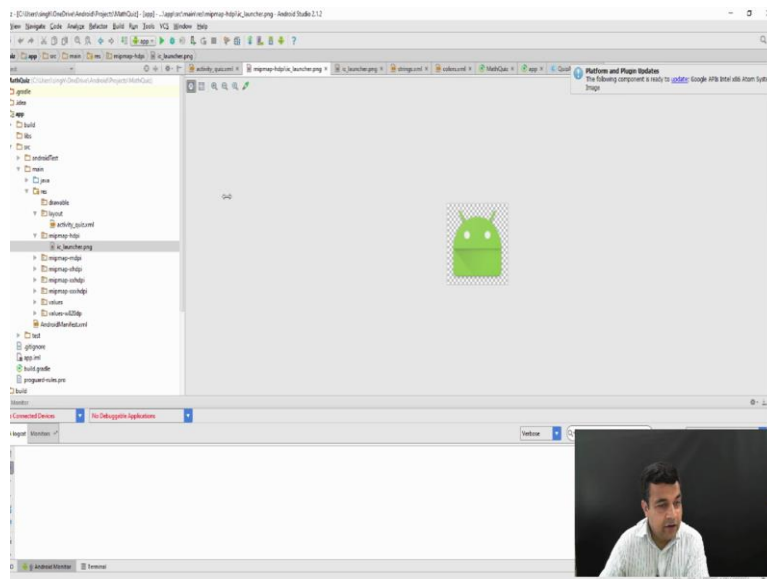
(Refer Slide Time: 18:38)



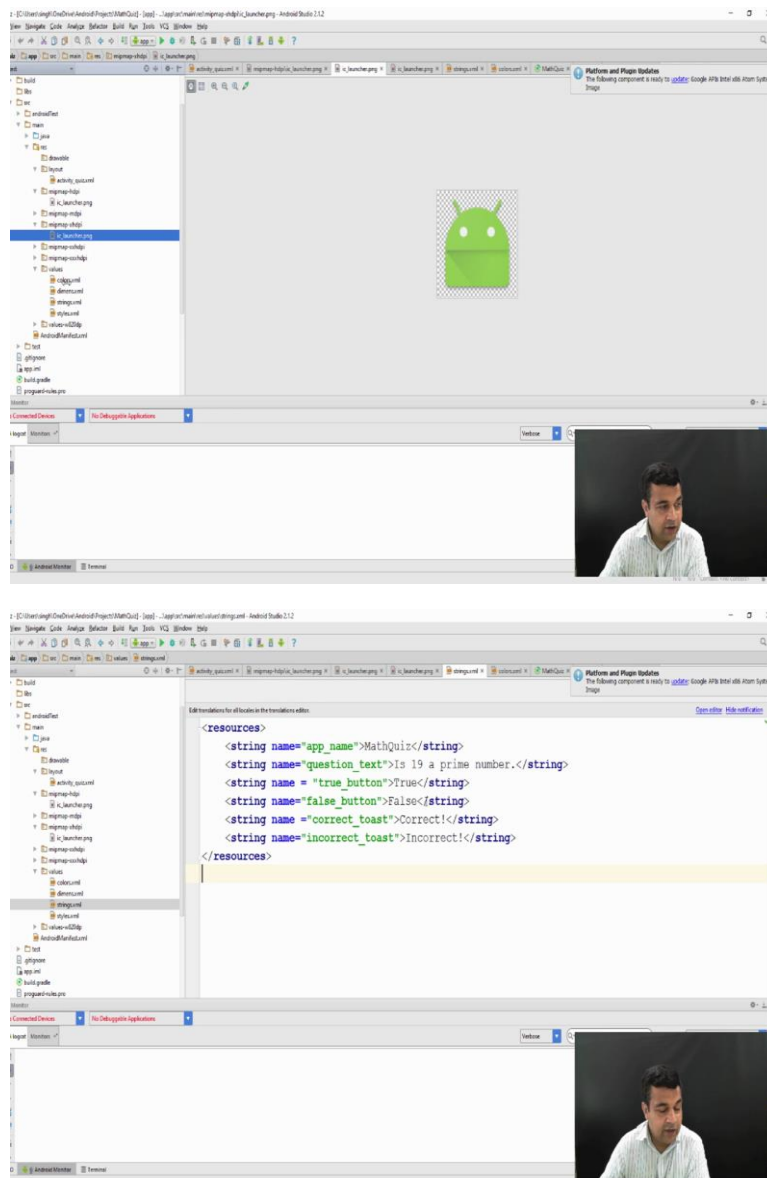


Inside layout there is a activity quiz.XML which is giving us text version of XML and also design version. And then we have different midmap directories with respect to different resolution.

(Refer Slide Time: 19:03)







So each mipmap directory has the launcher icon. So yes here is the familiar default android launcher icon and it is available into different densities. Depending on the density of your phone screen the right launcher icon is displayed. After that we have our values folder. Which is different other XMLs lets open the strings and as you can see all the strings that (( ))(19:27) are defined here.

So this was a quick revision of android studio and android project one more time. In fact this is our last time that I am introducing you android studio. And now and onward I will assume that you have installed android studio and you are ready to program. So for the next classes we will only be using concept which are advanced and will not be explained in harder side of android studio or start an android studio. Thank you!