O que é Flutter?

É um kit e estrutura de desenvolvimento de software feito pelo Google para criação de aplicativos móveis 2D, se você quiser qualquer tipo de jogo 3D, que seja executado em Android ou IOS não é o que você deve estar aprendendo agora.

hi guys my name is Aria and I welcome

00:12

you all to this flutter tutorial now

00:14

Before we jump into the video let me

00:16

just go ahead and give you guys a brief

00:17

introduction to the topics I intend to

00:19

cover today so first of all we will be

00:22

talking about mobile application

00:23

development and how it is done what it

00:25

is then we will look into the spotlight

00:28

matter for today and that is flutter so

00:30

we will be discussing what exactly is

00:32

flutter and why you should learn it

00:34

moving ahead we will be discussing the

00:36

architecture that makes a flutter the

00:38

types of widgets that flutter uses and

00:40

half ladder compiles in the end I will

00:43

show you guys how to install flutter on

00:44

your windows system and then we will be

00:47

writing a very simple application using

00:49

flutter itself okay so let's get started

00:52

so the first topic for today is the poll

00:54

of mobile application development now as

00:57

you guys know the world today is kind of

00:59

divided into iOS and Android now iOS and

01:02

Android take up around 99 percent of the

01:05

market share and comes to mobile phone

01:07

applications and mobile phone systems

01:09

now they both are actually rival

01:13

companies and as you guys know Apple or

01:16

iOS is developed by Apple and Android is

01:19

developed by Google but the story is not

01:21

the same for a software developer who is

01:24

targeting mobile applications because

01:26

for a software developer when he

01:28

develops an app and he wants it to be

01:30

successful he would want it to run on

01:33

port systems be given Android and iOS

01:35

take up 99% of the shares so what do you

01:39

want to do is you want to write an app

01:41

that will run on both systems and that

01:43

is pretty much prevalent in today's

01:44

norms you see that there's Google Maps

01:47

for Android there's Google Maps for iOS

01:50

there's even Apple maps for Google and

01:52

Apple Maps for iOS there are a lot of

01:54

mobile applications that run on both

01:57

platforms and most of the times to

01:59

improve performance companies will try

02:02

and go for the native approach now if

02:04

you don't know what I'm talking about

02:05

when I say native approach you might

02:07

want to go and watch my introduction to

02:09

mobile application development video

02:11

which we talked about a p.m. which

02:13

the testing framework and the world of

02:15

mobile application development so just

02:17

to give you guys a gist we normally have

02:19

three kinds of applications web hybrid

02:21

and native now if you want an

02:23

application to be performance specific

02:26

and have no sort of performance issues

02:29

and should be able to access all the

02:31

hardware features and all the software

02:33

can be optimized to the tee so that is

02:36

the time when you go for a native

02:37

approach mostly a native approach costs

02:39

a lot of money and this is because you

02:42

need to hire two different teams to

02:44

write applications for Android and iOS

02:47

when they are taking the native approach

02:49

for example if you are writing a native

02:52

app for iOS you will be needing a swift

02:54

developer and if you want the same app

02:56

to actually run on Android you will need

02:59

a Java developer somebody who is

03:01

proficient at Android the bluffin so

03:03

basically what you are doing out here

03:05

you are spending twice the amount of

03:08

money for the same app just so that it

03:11

can run on two different application

03:13

platforms now this has become quite a

03:16

big problem for smaller industries and

03:19

startups where they want an app to run

03:21

on both platforms and they end up not

03:23

having so much money that is a time they

03:26

go for the hybrid approach now in a

03:28

hybrid approach you mostly have one code

03:30

base that works on both platforms has

03:33

given Android and iOS but normally the

03:36

performance of the app takes a hit in

03:38

this measure so this has become a

03:40

problem for most startups and smaller

03:43

companies because they would be spending

03:45

a lot of money on applications just so

03:48

that they can run on both platforms and

03:50

they have a bigger scope of a crowd to

03:53

approach to now this is mostly because

03:55

of three reasons so the first reason is

03:58

an SDK now on Android and iOS both

04:01

different SDKs or software development

04:03

kits are used for the development of an

04:06

application now a software development

04:08

kit is typically just a set of software

04:10

tools and development tools that allow

04:13

the creation of applications and

04:15

software packages with a lot of ease now

04:18

these are different from frameworks and

04:19

I will explain how frameworks are

04:22

different because framework contains

04:23

more than just an SDK now moving on

04:25

they have different SDKs

04:27

above that they have different

04:28

frameworks because they come from

04:29

different SDKs so you don't want to

04:31

really have one true framework so that

04:33

you can develop enough using that

04:35

framework and it will work on both

04:37

platforms so this is quite missing in

04:40

the industry today so this brings about

04:42

a general lack of uniformity in today's

04:46

mobile application development world so

04:48

if there's a developer who's starting

04:51

out in his career and he wants to

04:53

develop an app that runs on both Android

04:55

and iOS he will have to spend a lot of

04:59

time just developing the app itself

05:01

instead of doing the nativities like

05:03

maintenance and actually optimizing it

05:06

so even for those nitty gritties he is

05:09

going to spend a lot of time because

05:11

both the application platforms work very

05:14

differently for example in iOS Bluetooth

05:17

behaves differently then in Android

05:19

keeping these things in mind we could

05:21

say that there is a general lack of

05:23

uniformity in case of SDKs and

05:25

frameworks when it comes to mobile

05:27

application development so to solve

05:29

these problems as usual Google has come

05:32

up with a solution and this solution is

05:34

called flutter so let's take a moment to

05:37

discuss what exactly is flat enough so

05:39

this is the spotlight topics in today

05:41

and this is what our entire videos will

05:43

be based on so what exactly is flutter

05:46

well flutter is a software development

05:48

kit and framework that is made by Google

05:51

for the creation of 2d mobile

05:54

applications so if you want any sort of

05:57

3d game that will run on an Android or

05:59

iOS device flutter is not what you

06:02

should be learning right now you should

06:03

be learning something else on the other

06:05

hand if you are trying to create a 2d

06:08

application that runs on both iOS and

06:10

Android flutter is certainly the way to

06:12

go so flutter is a software development

06:15

kit and it is made on the language

06:17

taught so dart is the main language used

06:20

around which the framework is built so

06:23

the framework is basically a set of

06:26

general tools that help you with the

06:28

whole mobile application development

06:29

process to give an analogy of a

06:32

framework let's say you cook breakfast

06:34

every day in the morning now to cook

06:37

breakfast you need a lot of things you

06:39

need your

06:41

vegetables you need your proteins that

06:43

can be chicken or eggs and you need your

06:46

cooking oil and all this is normally

06:48

spreaded out in your house and if

06:50

there's also the process of actually

06:52

cooking that whole thing now suppose all

06:54

your breakfast items would come to you

06:56

in a box every day at your doorstep so

06:59

suppose your bacon came to you in a box

07:01

with a bottle of milk and you got a loaf

07:03

of bread

07:04

you got some butter and you also got

07:06

some vegetables to just throw on the

07:08

grill so when all of these things come

07:10

to you in a box

07:11

in a unified manner you can call that as

07:14

a framework now put the same analogy in

07:16

a mobile application development wave so

07:19

think of something like you need a

07:21

search bar so something to make a search

07:23

bar already exists in the framework all

07:25

you have to do is write down the piece

07:26

of code which is already there and your

07:29

search bar is just ready to go it just

07:32

makes the whole development process a

07:34

lot more easy now the main USP of

07:37

flotter at this moment is that you have

07:39

one single code base for both iOS and

07:42

Android devices and they both act as

07:44

native apps so acting as native apps is

07:47

not suppose the right way to see it they

07:50

are native apps they work natively on

07:52

Android and iOS and we will understand

07:55

how so first of all let's go and

07:57

understand while on flood or in today's

07:59

varied world now today we have a lot of

08:03

other frameworks so flutter is not the

08:06

only framework out there that does the

08:08

whole cross platform gimmick there's

08:10

also Cortland where you can write client

08:12

and server-side code using one language

08:14

unless and it is a lot like JavaScript

08:17

so what exactly is going for flutter

08:19

well the first thing is the whole motto

08:21

that one for all and all for one

08:23

so flutter allows you to develop and

08:25

create native applications for both iOS

08:27

and Android but as I just said this is

08:29

not the first framework to do it there

08:31

have been a few but they have haven't

08:34

had the success so what's so different

08:36

about flutter well from one Google is

08:38

using it and when you know Google is

08:40

using something you can say that

08:41

framework or software or tool has a lot

08:43

of credibility going for it another

08:45

selling point in the flutter is it's

08:47

easy learning curve so flutter has been

08:50

heralded as a very easy framework to

08:53

learn it

08:54

if you are a beginner that is starting

08:56

out his career in mobile development

08:58

Shalonda is definitely the way to go

09:00

because that will give you one framework

09:02

that you can use to build both iOS and

09:04

Android apps so just think about the

09:07

doors of opportunities that you are

09:08

opening for yourself you can apply for

09:10

iOS Jobs you can apply for Android Jobs

09:13

and you can do it just from the get-go

09:15

without knowning Swift or Java and

09:18

personally for me Java has always been a

09:21

paint loan and master because I find it

09:23

extremely systematic as a programming

09:26

language and if you don't want to go

09:27

through that whole hell of an experience

09:29

of learning Java you can just learn

09:31

flutter and use it on Android

09:33

development and on iOS development too

09:35

so on the other hand if you are a person

09:39

with some experience in mobile

09:41

application development it will only be

09:43

more easy for you to learn because it

09:45

says that you have some experience with

09:47

object-oriented programming and mobile

09:49

application development as a whole the

09:51

third point that is why you should learn

09:54

flutter is because in today's

09:55

competitive world

09:57

it makes your resume shine so somebody

09:59

who is actually hiring a developer for

10:03

his or her company so that he can build

10:05

an app now according to the norms of

10:08

tape the person might want the app to

10:10

run on both the platforms so if he sees

10:13

flutter on your resume you know that you

10:16

can write both apps for Android and iOS

10:18

and this will not take a hit on the

10:21

performance of the app so this only

10:23

makes your resume a whole lot more

10:25

brighter and shinier in the eyes of

10:27

somebody who is going to employ you okay

10:29

so those were three great reasons that

10:31

you should definitely try and learn

10:32

flutter now let's move on with our topic

10:35

of discussion today so the next topic of

10:37

discussion in this video is gonna be

10:39

fathers architecture ok so now let's

10:41

take a deeper look into how flutter

10:43

works and let's discuss its architecture

10:45

so first of all on the left hand side of

10:48

the screen we see is an app made using

10:51

flutter well is a Jif

10:53

it is basic let's imagine that it uses

10:55

flutter now when a app is made using

10:58

flutter it actually uses something that

11:01

is called as a widget tree so everything

11:03

that you see on the app or the app

11:06

itself can be considered as

11:08

now the other elements that go along

11:10

with the widget like the menu section

11:13

the select sections the different

11:15

buttons the drop-down menus all of them

11:18

also found visits so basically what you

11:20

get in the end is a widget tree now the

11:24

main app will be the parent widget for

11:26

everything else then it responds

11:28

different children and these children

11:30

can be of different types of widgets and

11:32

we'll get to that later but for now all

11:34

you have to understand is that whenever

11:36

app is made using flat L it is basically

11:39

a collection of widgets and it looks

11:41

something like this which is basically

11:43

called a widget tree the moving forward

11:45

one might think that widgets are not

11:47

exactly the best way to go around and

11:51

using for a design layout or for design

11:54

performance well Google actually

11:56

disagrees with that and they have also

11:58

given you way to actually embrace

12:00

platform differences using the widget

12:02

tree so when I mean platform differences

12:05

I mean the difference is that lay in iOS

12:08

and Android both now

12:10

well apps might perform the same way and

12:12

act the same way on both platforms how

12:15

do you actually interact with the

12:17

hardware services like the glue to the

12:21

speakers everything is completely

12:23

different so you have different api's

12:25

that are called and handled for these

12:27

different services and the best part

12:29

about flutters architecture is that it

12:31

has those less column blank spaces in

12:34

the framework so that you can put them

12:37

yourself and you can develop and have an

12:40

app that has complete control over the

12:42

applications and its behaviors or

12:44

regardless if it runs on Android and iOS

12:46

poor now let's take a look at the

12:49

different types of widgets that we have

12:51

so first of all the flutter has two

12:53

types of widgets so the first one is

12:55

called the stateful widget and the other

12:57

is called stateless widget so whenever

13:00

you are making an app you aren't going

13:02

to be using a combination of stateful

13:04

and stateless visits these are the two

13:06

kind of widgets and they form a majority

13:08

of the UI elements like search bars

13:11

buttons invisible boxes just boxes for

13:15

layout purposes everything everything is

13:17

either or stateful widget or a shtetl

13:19

search so let's go and see what exact

13:22

these two are and how they are different

13:24

from each other so stateful widget is

13:26

basically one that can be changed when

13:28

ever the app is running so it has the

13:31

ability to be changed dynamically so

13:34

this means stateful widgets are mutable

13:35

and can be drawn multiple time within

13:37

its lifetime so this could be something

13:40

like a weather card now a weather card

13:42

actually changes throughout the day if

13:44

you are using Google Google's Android

13:46

service called Google now which shows

13:47

you the weather in a quick-draw and you

13:51

can see that it is shown in a card like

13:53

fashion now if you are using this

13:57

feature every day you will see that the

13:59

weather gets updated more often than not

14:01

so how is that possible well if you are

14:04

using flutter that will be a stateful

14:06

widget because that is changing all the

14:08

time on the other hand stateless widgets

14:10

are the complete opposite stateless

14:12

widgets are immutable ones built and to

14:14

draw a stateless widget you will need to

14:16

create a new instance of that widget

14:18

itself so suppose let's say the card in

14:21

itself is a stateless widget because the

14:24

shape of the card is not really changing

14:25

while the content is changing so the

14:28

content can be put in a nested fidgets

14:31

inside a stateful widget and that can be

14:33

I'm sorry so I mean to say that when you

14:37

are imagining a weather card the outer

14:39

card could be a stateless widget as the

14:42

shape of the widget is not changing it

14:43

is only being drawn once while the

14:45

contents of the card could be in an

14:47

invisible stateful widget inside and

14:49

they can be mutable from time to time so

14:53

these are the two kinds of widgets that

14:55

you get with flutter and these are the

14:57

types of widgets that you were going to

14:58

play around with now let's take a look

15:00

into how flutter codes actually compile

15:03

into native codes now most of the people

15:06

when I say that flutter code actually

15:08

compiles in the native code get really

15:10

confused as to how this is exactly

15:13

possible so well let's take a look into

15:15

how flutter actually compiles it codes

15:17

so at the center of all the service you

15:20

have your dart code which basically

15:22

makes up your visits and the code of

15:24

your software or your program or your

15:28

application whatever you'd like to call

15:29

it is being used by the dart code or is

15:32

being used and being built on the dart

15:34

code itself

15:35

aside from that now after the dart code

15:38

you have the flutter API that sits on

15:40

the dart code now the flutter API

15:43

provides you with an ease of development

15:46

process this lets you create your

15:48

widgets your stateful widgets your

15:50

stateless widgets and besides how you

15:52

are going to shape up the whole UI in a

15:55

much more uniform and systematic fashion

15:58

so it basically gives you utility and

16:00

widgets

16:01

now this dart code is then compiled into

16:04

armed binaries which is both understood

16:07

by iOS and Android so basically you are

16:10

not going down and using a specific

16:13

up-and-down compiler to get different

16:16

codes for different systems what you do

16:18

is basically just compile them into

16:20

armed binaries which is understood by

16:22

both platforms and this is done by the

16:24

flutter SDK so this allows you to

16:27

actually write dart code that can run on

16:30

both iOS and Android and as we have

16:32

already discussed the flood or framework

16:35

gives you enough flexibility to actually

16:37

buckle all sorts of changes that might

16:41

be there in both platforms so now is all

16:44

about flutter and what it is and why you

16:47

should learn it now let's go ahead and

16:50

take a look at how you can install

16:52

flutter on your own system okay so it's

16:55

time that we look at how you get install

16:57

flutter on your Windows system so to

17:00

install flutter first of all open up

17:02

your browser and go to flutter on tab so

17:05

this is the site where all the news

17:07

about flutter is going to be available

17:08

you can go and see the docs out here you

17:11

can see the community and what it looks

17:13

like and it also has a whole get started

17:16

guide out here now out here you have to

17:19

go and click on get started and first of

17:22

all this will direct you to the

17:24

installation page now all you have to do

17:26

is click on Windows and this will come

17:30

and show you the system requirements and

17:32

what you need to do to actually install

17:35

flutter on your system so out here to

17:38

insult and run flutter your development

17:40

environment must meet these minimum

17:41

requirements so first of all we need

17:43

operating systems such as Windows 7 sp1

17:45

or layer Plateau and we need a disk

17:48

space of around

17:49

100mb and this does not include the disk

17:51

space for ide and tools and we get to

17:54

the ide part later and for tool section

17:57

we flutter depends on these tools for

18:00

being available in your environment so

18:02

first of all is Windows PowerShell 5 or

18:04

newer and the second one is git so to

18:07

install gate all you have to do is go

18:09

ahead and click on that link and that

18:11

should go ahead and show you your git

18:15

download page so I already have git

18:17

installed on my computer and the path

18:19

already set if you don't have that you

18:22

can go ahead and see that in a separate

18:24

YouTube video and how to install it set

18:26

of git for now I'm just going to discard

18:28

this so for now what we have out here is

18:31

we need git and we need Windows

18:33

PowerShell 5 so go ahead and install

18:36

those boat so let's read up on how to

18:38

get the flutter SDK so first of all we

18:41

need to download the flutter zip file

18:43

that is there so let's go ahead and

18:45

download that so let's download this and

18:48

that will be the flutter SDK so you see

18:51

that this is a following installation

18:53

bundle to get the latest stable release

18:55

of flutter Brother releases and only you

18:57

can go for the SDK archive so the second

19:00

thing that we need to do is extract the

19:02

zip file in place to contain flutter in

19:04

a decided station location it says that

19:07

we should not put it in a directory like

19:09

Program Files because that requires

19:10

elevated privileges and to locate the

19:13

flutter file we have to actually go

19:16

ahead and hit the flutter con solder pad

19:18

ok so let's go ahead and actually see

19:21

how we can do that so let's wait for

19:23

this thing to download up now on the ide

19:26

side of things I would recommend that

19:28

you go ahead and download Android studio

19:30

on your system downloading under studio

19:33

is pretty pretty simple if you have some

19:36

trouble downloading Android studio you

19:38

can go ahead and look at my appium

19:40

installation because during a PIM

19:43

installation we also go ahead and

19:45

install Android studio but just in case

19:47

you don't feel like watching a new video

19:49

for this particular thing all you have

19:51

to do is go to developer.android.com

19:53

slash studio which can be easily reached

19:56

by just googling android studio on your

19:58

favorite browser now once you open the

20:00

page the first thing that you will see

20:02

is this pic

20:03

huge Android studio download button all

20:05

you have to do out here is agree to the

20:07

terms and condition which nobody

20:09

probably reads and then all you have to

20:11

do is click on download now I don't

20:14

really want this file because I already

20:16

have it installed so I'm going to

20:17

discard that but for now let's

20:19

concentrate on the spotlight for today

20:21

and that is flutter so as we can see we

20:24

have this downloaded and as you guys can

20:27

see I had already previously downloaded

20:28

it and this is the one out here and that

20:32

can be unzipped into this file once you

20:35

have unzip the file you will get this

20:38

folder called flutter windows version

20:40

1.2 point one stable in this what you

20:43

have to do is go ahead wait aaseesh

20:46

decompress this and show it to you so

20:48

all you have to do is use your

20:50

decompressing software I have 7-zip and

20:53

I'm going to use that so let's just

20:55

decompress them and see what we get so

20:58

let this process run and I'll be back

21:01

with you guys as soon as this process

21:03

has finished

21:07

okay so now that we have extracted our

21:09

folder we see that we have a flutter

21:12

folder in the folder that we extracted

21:14

it to and according to our instructions

21:17

that we read on the installation page

21:19

what we are supposed to do is go ahead

21:21

and click this now out here you see a

21:24

bunch of other instructions so use the

21:26

console below or this message to

21:28

interact with the clutter come on so

21:30

what you want to do out here is run

21:31

flutter docto to check if your system is

21:33

ready to run flutter apps and once that

21:36

is done all you have to do is run

21:38

flutter create an app name so that's

21:40

very simple now if you want to run

21:42

flutter commands from any command from

21:44

or partial window all you have to do is

21:46

go ahead and set your pot so the setting

21:49

apart all you have to do is go ahead and

21:51

put this folder into any place that you

21:54

would want now I could just put this

21:57

folder here but what you have seen is

22:00

that I normally put my folders in my eat

22:03

ripe so out here if you see I have the

22:05

flutter folder and all you have to do is

22:07

go into the bin folder and copy this out

22:09

out here copy the address go into this

22:12

PC click on properties and click on

22:16

advanced system settings and environment

22:19

variables and only how to do out here is

22:21

going to part under system variables so

22:24

definitely found out here going to edit

22:27

and all you have to do is put in the bin

22:30

folders address out your so as you guys

22:32

can see I have already done it here and

22:34

you can do this yourself by actually

22:37

coughing down the address so that will

22:39

actually put your flutter bin folders

22:42

address into your system variable so

22:44

that your system understands that

22:46

flutter is a command that should be

22:47

understood by your system ok so that

22:50

sets up for setting the environment now

22:53

all you have to do to see if you are

22:55

ready to run flutter or flutter

22:58

applications all you have to do is go

23:00

ahead and your command prompt and all

23:03

you have to do is go ahead and run the

23:06

command flutter dr. now as you guys can

23:09

see out here when I run flash dr. Oh God

23:12

I'm running the same thing again so if I

23:14

go and run flutter dr. it gives me this

23:17

doctor summary so this doctor somebody

23:20

tells us

23:21

required things that you might require

23:23

for running a flutter app so as you guys

23:25

can see I have the flat o channel and

23:28

the flutter SDK installed it's all to

23:30

the Android to chain I do not have an

23:33

Android license status because that's

23:34

unknown

23:35

I have Android studio installed and

23:37

there's no connected devices so the

23:39

doctor has found two issues in this

23:41

category but those issues can are not

23:44

really very major but all you need is

23:48

mostly a droid studio and the flutter

23:50

SDK so if you didn't have anything other

23:53

than these doctor would tell you what

23:55

other things you require for flutter to

23:57

be actually running on your system so

23:59

another thing you can do is create

24:01

flutter apps using the flutter come on

24:03

so let's go into our desktop so once we

24:06

are in the desktop we can go ahead and

24:07

say flutter create let's say demo so

24:11

we're gonna name our application demo

24:14

let's go ahead and hit enter and this

24:16

will go ahead demo is not a valid dart

24:19

package name to use lower score so let's

24:22

see we can do demo and it should go

24:25

ahead and create our flutter app that

24:28

will be ready to run now we will be

24:31

editing this out ourselves and we are

24:34

gonna make our first app using this so

24:37

on the Android studio side you also want

24:39

to make sure that you have a few plugins

24:42

that are there for your flutter

24:44

application and your flutter development

24:46

to go seamlessly so let's go ahead and

24:49

look at those so once

24:51

Android studio loads up you will see

24:53

that I have a separate option that says

24:56

I can create a new flutter project so

24:59

I'll teach you guys how you can actually

25:01

get that so let's just wait for this to

25:04

load up

25:06

okay so it seems like my android studio

25:09

is loading up a separate check so let's

25:12

go ahead and see that so let's close

25:14

this tick box and it's also close is

25:19

okay so we don't want to actually exit

25:22

out nurse studio but what we want to do

25:24

is open so what we have to do is go

25:27

ahead and go to our poll go to desktop

25:31

out here you will see that there is a

25:34

demo app and if we open that lets open

25:37

it in this window so this app is a basic

25:41

app that comes pre-loaded whenever you

25:43

make a flutter app now what I want to

25:45

draw your attentions to is you have to

25:48

go into file and then into settings and

25:50

out here you have to download a few

25:52

plugins before you actually go head and

25:54

start developing your flutter apps so

25:56

until you go into plugins and you're

25:59

going to browse repositories and out

26:01

here all you have to do is go ahead and

26:04

search for flutter so out here you see

26:08

that this is flutter and you can go

26:10

ahead and download that now I have

26:12

already downloaded that so it's not

26:14

showing me a green download button so it

26:16

will show you something like that and

26:18

other than flutter you also have to

26:20

include dart so I already have the dart

26:24

plug-in installed so it is also not

26:27

showing me that install buttons but if

26:30

you were to actually install it it will

26:32

show you a green plugin button like this

26:34

I mean a green install button like this

26:36

so all you have to do is go ahead and do

26:39

the repositories and install the plug-in

26:41

for dart and for flutter okay so that

26:45

would set up your Android studio for

26:47

writing your first flutter application

26:49

so now let's go ahead and see how our

26:53

first flutter application actually looks

26:55

like so out here if I actually go into

26:58

tools in the end of abd so let's go into

27:01

EVD manager and let's start up our

27:04

virtual device so you need an emulator

27:08

or a virtual device to actually run and

27:10

test your apps and we are going to be

27:12

using Android studio just because it

27:14

gives us this whole EVD feature now

27:17

let's go ahead and turn this on

27:19

now what do you see out here let's go

27:21

ahead and see this is the main door dart

27:23

file and if you're going to project and

27:27

let's go into packages so demo and we

27:30

have this made not hard file so this is

27:32

the part where your code is written for

27:34

most of your apps for today's simple

27:36

application we will be writing our code

27:38

in the main the dart file and this is

27:41

not going to be very complex application

27:43

it is just to make you apply what you

27:46

have learned today that is how you can

27:48

install flutter and make a couple of

27:50

widgets like a stateful widget and the

27:52

stateless widget and see how flutter

27:54

runs both on Android and iOS so let's

27:57

see if our emulator has started okay so

28:00

we see that the boot logo has come up

28:03

and as soon as let's just wait for the

28:05

you wive startup

28:07

okay so now that our emulator has

28:09

started out what you see out here is an

28:11

red phone that is being emulated right

28:13

now so this is a pixel to model and the

28:16

model doesn't really matter out here as

28:17

much as the API version so we are

28:20

currently running Android eight out here

28:22

so my purpose of actually puffing up

28:26

this emulator is to show you how a

28:28

flutter app looks like when it is

28:30

actually run so what we can do out here

28:32

is go ahead and press this Run button

28:35

and this will go ahead and out here even

28:39

if you will go to the terminal section

28:41

out here all you could do is go ahead

28:43

and say flutter run and this would start

28:46

up your app on your emulator so as you

28:50

guys can see this will give you a

28:51

summary on the processes that it will

28:53

run while actually starting up the app

28:56

and let's see what goes on in the app

28:58

right now so let's just see what's gonna

29:01

happen now this is not to be a complex

29:04

app and people then the palm is up just

29:06

to show how flutter works so today we

29:09

are going to be actually making a random

29:12

word generator with somewhat of an

29:14

infinite scroll but before that let me

29:17

actually wait for this run command to

29:20

actually be executed

29:23

okay so that seems to have failed

29:25

because our Gradle is still being

29:28

actually being assembled now that our

29:31

table has been assembled

29:33

I see the process out here has actually

29:35

stopped so let's see so there seems to

29:38

be in some sort of error we should

29:41

actually go ahead and resolve that so

29:43

let me just go ahead and run this come

29:45

on once again

29:48

okay so as you guys can see our screen

29:51

has gone fight on our emulator at this

29:53

moment so let's just wait for the app to

29:55

pop up

29:57

so as the summary is going it's

30:00

currently installing the build and apk

30:03

to our system

30:07

so our system UI isn't responding at

30:09

this moment so we can just go ahead and

30:11

click the wait option and that should

30:12

normally go ahead and dismiss da for now

30:15

let's see if our app pops up now this is

30:18

generally taking a lot more time because

30:20

I haven't actually given so much memory

30:23

to my hack simulator so let's see if it

30:26

actually pops up there are me actually

30:28

going and fiddling with the memory

30:30

settings so seems like it will pop up as

30:32

it's syncing the files

30:37

okay so it seems that our flutter app is

30:39

running somehow but this is taking a lot

30:43

more time expected to actually load up

30:49

okay so it seems like I have ended the

30:52

connection to the app right that also

30:54

should have so let's go ahead and do a

30:58

flutter run again

31:03

okay so as you guys can see after

31:05

running fluttered run on our terminal

31:08

we have popped up the flutter app or to

31:10

our emulator now as you guys can see out

31:12

here this is a very basic app and if I

31:15

go ahead and press this button out here

31:18

why isn't this getting pressed if you go

31:21

ahead and press that button out there

31:22

the counter of the button actually goes

31:25

up and we can see that being represented

31:27

in the code somewhere out here so we see

31:30

that a counter has been set out here

31:32

which is int counter equals to zero and

31:35

every time we are going to set a state

31:37

and that is going to be counter plus

31:39

plus now what we want to do out here

31:41

let's see now let me show you a cool

31:43

thing about flutter and that is called a

31:46

hot reload now if I were to go to this

31:48

counter part and do equals to counter

31:53

plus two now out here I could just go

31:57

ahead and press R and that will actually

32:00

go ahead and hot reload my entire app

32:03

now let the hot reload commands now if

32:07

you see I pressed the button once and

32:09

went up to 14 and if I press the button

32:12

again goes up to 16 so as you guys can

32:14

see changes in your code with the hot

32:16

reload are reflected almost

32:18

instantaneously in your app so that's a

32:21

great thing about flutter now the final

32:23

part of today's presentation is that we

32:26

are going to try and install and run our

32:29

own simplify our application so do to do

32:33

that let's go back to our Android studio

32:35

project and we are going to write our

32:39

own piece of poop so this piece of code

32:41

or as you might say application is going

32:44

to be an application for generating

32:46

board so we are basically going to build

32:48

a word generator using flutter today so

32:51

what we are going to do first is

32:53

basically let's remove all this code

32:56

from your first one right now what we

32:58

want to do is we are going to write our

33:00

own piece of code so for that just

33:02

follow along and I'll try and explain

33:04

what I'm doing because this is a very

33:06

introductory video to flutter and if

33:08

this video gets a lot of appreciation we

33:11

are going to produce a flutter playlist

33:13

which will actually go into the index of

33:15

the entire framework

33:17

so just hang on and try to follow along

33:19

as to what I'm doing so first of all we

33:21

are going to actually create a flat or

33:25

half now so the first thing that you

33:28

want to do is going to mean dot dart

33:29

under the lip folder and delete all the

33:31

code that is that no we are gonna be

33:34

making a bow generator so I'm gonna be

33:36

driving you down the code line by line

33:38

so first of all we need to import a

33:40

certain package and that package is

33:42

called and this is how you import

33:44

packages in flutter all you have to say

33:47

is package and the first import package

33:49

is trackage slash emotional dot dot and

33:51

since I have my plug-in ready that is my

33:54

shuttle plugin and the dart plugin this

33:56

is being so much easier now what we want

33:59

to say is void main' and you want to

34:02

open and close the bracket and then we

34:04

want to say run app so run app and let's

34:08

say my app fine so this is the basic

34:12

command that you would put into any

34:14

flutter application so that your app get

34:17

started now all you have to say is class

34:20

my app and the same way that you spent

34:23

it in the second line and you're gonna

34:27

make this extend so you make it extend

34:30

something by saying extends and you say

34:32

stateless so this is going to be a

34:35

stateless widget so we can go ahead and

34:36

select that and we say stay to this

34:38

widget and now what we want to do is

34:40

open our parenthesis and now when we

34:44

want to say is something important we

34:46

want to say we this will override every

34:49

single widget that is built into this

34:51

class the first widget that we are going

34:53

to build is going to be with the build

34:55

function so we say widget build and what

34:59

you see out here is a basic widget that

35:02

has been built for us but what we want

35:05

to do is not exactly gonna be so basic

35:08

so you want to return null and that's

35:11

not what we want to do so what we want

35:14

to do is return our material so we

35:18

wanted to turn a material app so to

35:20

determine material app you could say

35:22

something like a title of the app so

35:26

title would be let's say welcome to

35:29

welcome to my first application okay now

35:35

over here we also want to set up the

35:37

home so for home all you want to say is

35:41

we have to define a home and once we

35:44

define a home we can put a scaffold is

35:47

now a scaffold it is just an element you

35:49

learn in future videos and I'll go into

35:53

the details of all these elements so

35:54

what do you want to say is this is our

35:58

scaffold now what we want to do is build

35:59

an app bar now the app bar is also going

36:03

to be built with the app bar function

36:05

and now you say app bar and all you have

36:07

to do is put in the title of the app bar

36:10

in the title we could give about them

36:13

text so we could say something like text

36:15

and we could say again let's put

36:20

something like welcome to my first

36:23

application now that should put up a

36:27

title and now in the center I would like

36:31

to also show a hello world so outside of

36:35

the app bar so this is the app bar now

36:37

we come into the scaffold so we go to

36:39

the app bar and we want to create a body

36:41

now our body is gonna be in the center

36:44

so let's see Center and center should

36:47

pop up and what do you want to say in

36:50

the center we want to treat a child

36:51

remember everything is a widget so what

36:54

we want to say is this is going to be a

36:55

child fidget and it will show a text and

36:59

in this piece of text what we want to

37:02

say is hello world so once that piece of

37:06

text has actually been done you want to

37:10

go into material app and we want to put

37:12

a semicolon out there so that everything

37:14

is properly let's see this ends here and

37:17

this ends here so what we want to do now

37:20

is go into our terminal and press a hot

37:23

real now let's go into our app and as

37:26

you guys can see we have a nice little

37:28

scaffold going out here and it says

37:30

welcome to my first application and if

37:32

you were to zoom in out here it says

37:35

hello world so as you guys can see we

37:37

wrote a little bit of code and this has

37:40

started transforming our application

37:42

into some

37:43

that we want so what we have out here is

37:45

hello world now the next step what we

37:48

want to do is run the app so we've done

37:50

that so the scaffold widget from the

37:52

material library provides a default app

37:54

bar title and the body property that

37:56

holds the widget tree for the home

37:58

screen so the widget tree can be quite

38:00

complex at times but for now we're going

38:03

to be keeping it real simple

38:04

now the widgets main job is to provide a

38:06

bill function method and that is a build

38:08

method that we just solve and that

38:10

describes how to display the widget in

38:11

terms of other lower-level widgets and

38:13

the body for this example consists of a

38:15

center which is containing a textile

38:17

widget and the center widget aligns

38:19

itself with the widgets subtree to the

38:21

center of the screen and that says hello

38:23

world now the next thing that we are

38:25

going to learn is to use an external

38:28

package in this application that we are

38:30

building so to do that first we have to

38:33

go to the pub spec dot gamma file so

38:35

let's go ahead and open that and out

38:37

here you see all sorts of dependencies

38:39

and stuff going on so inside the

38:42

dependencies part what we wanna add is a

38:45

line so let's add that line so this is

38:49

going to say English words and the name

38:53

of the package is basically English

38:55

words and we want the version 3.1 points

39:00

here oh so well that's that and that's

39:04

all we want for now now let's go back to

39:06

our main the dart file okay so as you

39:09

guys can see we have changed our

39:11

dependencies and our IDE already tells

39:14

us we can get dependencies so let's go

39:16

ahead and click get dependencies ok so

39:19

it seems that all our dependencies have

39:21

been fetched so what we want to do next

39:24

is import our new package stuff we just

39:26

add it to our dependency and we can do

39:28

that by actually going and typing the

39:31

import come on again and we want to say

39:34

is so it's single quotes and package

39:37

English words English word starter as

39:39

you guys can see sensitive added amount

39:41

of dependencies this thing pops up so

39:44

that is the next thing that you have to

39:46

do

39:47

Oh another change stuff you want to do

39:49

before actually doing our next hot

39:51

reload is let's see if a horror word is

39:54

actually loading up so what we want to

39:57

say up here is in our visit build part

40:00

so let's go into our vision field and

40:03

what we want to say here is we want a

40:06

final word pair so let's see final and

40:09

so final is basically like a constant so

40:12

final word and let's see but let's call

40:16

it word pair because we are going to be

40:20

generating a pair of oars it's going to

40:22

be equals to so this is the function

40:26

that comes along with the English word

40:29

slash English words art art so English

40:31

word pair dot random so word pair dot

40:34

random is a function that we are going

40:37

to go for now in this child text part

40:39

what you want to do is say let's remove

40:42

hello world and we do not need the

40:45

semicolons anymore I mean the single

40:48

quotes now if you want to say is word

40:50

pair chart as path Lee case so you want

40:55

the Pascal case so as Pascal case

40:57

basically means as every word will

41:00

basically begin with a new capital

41:01

letter so it's kind of like title case

41:03

you can look up what the case means and

41:05

that's for your grammar lessons but for

41:07

now let's go ahead and do this hot

41:10

reload out here so let's go ahead and

41:12

press R so let's go into our terminal

41:15

and let's go ahead and press R so that

41:19

seems that there seems to be an error so

41:22

there seems to be there we need to put a

41:25

semicolon off here and let's try hot

41:28

reloading again

41:31

okay so we need a semicolon after these

41:34

two so that was my mistake

41:36

no let's go ahead and hot reload that

41:39

too so going to the terminal part and

41:42

press R and that will perform a heart

41:44

reload so if we go ahead and go into our

41:47

app now what we see out here is two

41:49

little words so out here it shows that

41:51

means you swim into it so out here if

41:55

you see it says the word web mess now if

41:58

I were to go ahead and hot reload again

42:00

it will show me a new word every time

42:03

now this time it says to let us crush so

42:06

what we have done is basically remove a

42:09

added new dependency and external

42:11

package and use it in our app okay so

42:14

that kind of covers one aspect of what I

42:17

wanted to show you guys as to how you

42:19

can import new packages into your

42:22

existing applications now the next thing

42:25

that we will be doing is adding a

42:27

stateful widget to our application so

42:28

stateless widgets are immutable meaning

42:31

that their properties can't change as

42:32

all finally your final values are final

42:35

so stateful widgets maintain state that

42:37

might change during the lifetime of the

42:39

widget now implementing a steeple which

42:42

it requires at least two classes one a

42:43

stateful widget class that creates the

42:46

instance and the state class

42:48

so the stateful widget class is itself

42:50

immutable but the state class persists

42:52

over the lifetime of the widget so in

42:54

the step we will be saying how we can

42:56

have a stateful widget so let's go ahead

42:59

and see that so first of all we need to

43:01

create a minimal state class so we need

43:04

to add the following code into our main

43:06

door dart file so first of all we will

43:08

be creating a new class so out here what

43:11

you want to say is let's see where our

43:13

class ends or a class ends there so we

43:16

want to add a new class and this class

43:17

is gonna be called random word States

43:22

and it is going to extend so extends and

43:26

it is gonna be the state of random words

43:30

so we write it like this so random words

43:35

so that's how you do that now out here

43:38

let's just keep this empty for now so

43:41

like that so

43:43

we have created a class called class

43:45

random word state which extends this is

43:47

the state of random words

43:49

so notice something about the

43:51

declaration it says date and then random

43:53

vote so this indicates that we are using

43:55

the generic state class specialized for

43:57

use with the random words most of the

43:59

apps logic and the state resides here it

44:02

maintains the state of the random words

44:04

widget so this class saves the generated

44:06

word pair which grows in fluently as the

44:09

users crows and the favorite word pairs

44:11

in if we add that feature of favoriting

44:14

of WordPad now we need to add the

44:16

stateful random words wish it to the

44:18

main dot dot files so the random word

44:20

which is thus a little else beside

44:22

creating its date so what we have to do

44:25

is create a class called random word so

44:28

let's go ahead and do that it's a class

44:30

random words and this will extend

44:34

stateful widgets so extends just a full

44:40

widgets so out here we want to open up

44:43

the class and we want to say a then it

44:46

overrides so at the rate override and

44:49

what does it have to / a so random word

44:54

state so random word state and this will

44:58

create States so let's see so we create

45:02

the state with the create state function

45:04

so create state and this will be new

45:08

random word state so new random words

45:13

State fine so this we have created a new

45:19

instance of the state that we created

45:21

right out here so that's for the

45:23

semicolon after that now after adding

45:25

the state class the ID complains that

45:28

the class is missing a build method

45:30

let's see you see out here so we

45:33

actually have to add the build method so

45:36

out here we only want to say is widget

45:39

build

45:40

so this is actually missing a tilt

45:44

method so missing comforted

45:46

implementation of state dot bill so

45:48

Pomona a dot here is let's see first we

45:52

override and we want to bill the widgets

45:55

a widget build and while this is

45:58

returning now we don't really wanna do

45:59

that so widget build and build context

46:03

so that will return a context object so

46:06

what we want to say is final word pair

46:09

equals to WordPad or random so what kind

46:13

of random and only one do is return text

46:18

that says word pair as passive case so

46:23

word pair Todd as Pascal keys so let's

46:29

not forget to put our semicolons in as

46:32

semicolons are necessary in dart as a

46:34

language now since we have added this to

46:38

our state out here what we want to do is

46:42

remove this similar commands in our main

46:45

file so if you want to remove from here

46:47

is the final word pair so let's go ahead

46:51

and remove that

46:53

so let me just quickly select this and

46:55

leet it out and we also want to actually

46:59

go ahead and remove this child out here

47:03

now instead of this child giving a text

47:05

what we want to see is it will give us

47:08

random words so random words so once we

47:14

do this we should actually restart the

47:16

app so let's go ahead and hit a hot

47:19

reload and let's see the changes that

47:22

have been in our app now it's basically

47:24

the same thing but all we have done is

47:26

put in a stateless widget so each time

47:30

you save the app and you again hot

47:32

reload you all got to see another word

47:35

go in on there so I'll here you see pod

47:38

keys and you saw our different app a

47:41

different word the previous time so now

47:44

it's time to create an infinite

47:46

scrolling list view so in the step

47:48

you'll expand the round of words state

47:50

to generate and display a list of word

47:52

filing so as the user Scrolls the list

47:54

display in the list view widget flows

47:56

infinitely and the ListView built a

47:57

factory constructor allows you to build

47:59

a list like that and this is basically

48:01

called lazy scroll so what we want to do

48:04

is add a suggestions list to the random

48:06

word state class so firstly let's go

48:09

there so out here what you want to do is

48:12

add a bunch of suggestions so we can do

48:16

this before the override part so what do

48:19

you want to stay out here final and

48:21

let's see suggestions now you can name

48:24

available whatever you want but I

48:26

normally you'll name them with an

48:28

underscore because that's how normally

48:30

variables are named in the industry so

48:33

what do you wanna see is word fair and

48:36

you want to add an area out here so that

48:39

is how you would say it and let's see

48:44

and you wanna our another variable or

48:46

bigger font so that we can make the font

48:49

size bigger so let's see bigger font so

48:53

out here what you wanna say is constant

48:56

and text

48:59

style if I was not wrong per se style

49:02

textile and you want to say a font size

49:05

is let's see font size is 18 point O so

49:12

that does that so let's go ahead and end

49:15

I with a semicolon so next you will add

49:17

a build suggestion so let's go ahead and

49:21

see how we can add a bit suggestion

49:23

function to the random words tape class

49:25

so out here what you want to do is add

49:28

another widget so we want to say widget

49:32

and we want to build suggestions so

49:37

let's call this pill suggestions and

49:40

let's go ahead and open that up

49:42

so if anything is grayed out don't be

49:45

worried it just means that it hasn't

49:47

been used so let's go ahead and do that

49:50

so build suggestions so what we want to

49:53

do is return on this view dot builder so

49:55

return list view dot builder now in

50:01

Spencer what we want to do is go ahead

50:03

and create a list a few want to return

50:06

so first of all we need to add some

50:09

padding so for padding we are gonna say

50:11

constant edge inserts so edge inside is

50:15

basically a little cover that up right

50:18

now so edge it's set so we want to go

50:22

for the edge inset at all and let's see

50:24

so basically it will give padding on all

50:27

sides so basically what you would do in

50:30

HTML and CSS if you remember if you were

50:33

to put for different values of padding

50:35

that would put in top right bottom and

50:37

then left this is basically a shorter

50:40

way to do that in dot and all you have

50:41

to say is amount of padding you want so

50:43

we want 16 padding now what we want to

50:46

say is when item build up so we want to

50:49

build an item so we need an item builder

50:52

so say item builder and out here you put

50:56

in context and I so let's see context

50:59

then I out here what you want to do

51:02

after this is remove this open this up

51:05

in item builder what do you want to say

51:08

is if

51:09

so this is how you do if statements in

51:11

dart if I dot is odd so this will check

51:16

if it is an odd so what you want to do

51:19

if it's odd is return or divide oh and

51:22

if that is not the case what do you want

51:26

to do is go ahead and put in the final

51:29

index as I and then scraggily backslash

51:35

and - okay so that should do it

51:39

let's see that is the final and if then

51:42

we also want is one another a statement

51:44

so if index is greater than equals two

51:49

suggestions dot length so suggestions

51:54

dot length

51:58

so if it is greater than the suggestion

52:00

so that means a few I'll stops if you

52:02

are going more then what is meant to be

52:04

scrolled you want to add some more

52:06

suggestions to it so we can do that by

52:09

going suggestions equals 2 or rather

52:14

just say suggestions dot add all and

52:19

let's see generate word pairs and let's

52:25

see we need to put a take function so

52:27

take and 10 so we won't take them so

52:32

that should finish this part of the if

52:34

and this part of the if was just

52:37

returned divider and after that what we

52:39

want to do is return the build row so

52:43

we're going to do is return build row

52:47

was not popping up so let's see return

52:50

build row and this will be having

52:54

suggestions of index basically this is

52:58

how you formulate an array and you're

53:01

manipulating it to actually form

53:03

infinitely scrolling view so don't

53:06

forget to put your semicolons so this

53:09

should be a semicolon here and here okay

53:12

so that returns an item builder and this

53:17

is the list view builder and we have to

53:19

put a semicolon here too and that ends

53:22

our class okay so let me just explain a

53:25

bit so the item build a callback is

53:28

called once per suggested word padding

53:30

and places each suggestion into a list

53:33

I'll row or even rows the function adds

53:36

a list I'll row for the word padding and

53:38

for the all rows the function adds it

53:40

divided which it to visually separate

53:42

the entries note that the divider may be

53:45

difficult to see on smaller devices now

53:48

we add a one pixel high divider widget

53:50

before each row in the list view and the

53:53

expression I and the squirrel backslash

53:55

two divides I by two and returns an

53:58

integer so what you see out here is

54:00

basically a division operation and it

54:03

returns a literate in teacher as a

54:04

result for example one two three four

54:07

becomes 0 1 2 2 so 1 by 2 becomes 0 2 by

54:11

2

54:12

1 3 by 2 becomes 1 4 by 2 becomes 2 and

54:15

5 by 2 becomes 2 so this calculates the

54:17

number of world pairings in the list

54:19

view - the divided revision so if you

54:22

reach the end of the available word

54:23

pairing then it will generate 10 more

54:26

and then add them to the suggestions

54:27

list so now what we need to do is add a

54:30

build row function to the random word

54:32

State so what we need to do now right

54:34

now is go back to our steepest widget

54:38

and what we see out here is that we have

54:40

returned a scaffold so in the scaffold

54:44

we are saying the title and what we need

54:46

to do is change the body so we change

54:49

the body by saying let's put this body

54:52

out and the body will be returned by

54:54

pill suggestions so build suggestions

55:00

okay so that I think that our ListView

55:02

builder what we need to do next is go

55:05

ahead and put in the build role function

55:09

so let's see how we can do that so first

of all what we have to do is go ahead

and put in the build row which it so we

do that in the random state class so

random word state that part so we only

have to do is say widget we say widget

and it's going to be called build row so

out here what we say is word pair so we

are getting a word pair and this is

going to be called pair for us and let's

open up the widget class so ultimately

we want to return a list style and in

this style all you wanna say is title is

gonna be the text and so in this text we

want it basically make it back encase so

we say pair chart as fast we case and we

also say it style is basically bigger

font so remember the bigger font that we

had set up so we're going to be using it

here so out here all you want to do is

it or the style so you want to put a

semicolon out here and that bears our

row so in the random so with build row

this is how you are going to do it now

in the random word state class you have

to update the build method to use the

fill suggestions so we want to use our

build suggestions in the build part of

our random word generator stateless

class so what we want to do is go ahead

and put it in the following piece of

coal in our build part so let's go ahead

and find the build you chose random word

stator this is the build part so let's

go ahead and leave this so what we want

to do is basically return a scaffold and

a scaffold is as I just said it is an

app bar so it basically has space for

your app bars let's go hand nether now

so app bar and in the app bar we want

specify the app part part and we want to

say it has a title and title is gonna

say text and let's just call this word

generator so word generator so that is

gonna generate our words and we also

need a body so we need a body so the

body will be and build suggestions so

you're going to be doing it for the bill

suggestions part so that is why we did

this entire process and after that you

only have to is in do scaffold now in

the my app class update the build method

by changing the title and shading the

home to be random words so all you have

to do is go to class my app and we'll be

changing a bunch of stuff so first of

all we need to remove the title and the

home and let's improve the title out

here and let's call it word generator

and we are gonna update the home for

retired random words okay so let's see

the return random words so as you guys

can see we have simplified our my app

part at least with a lot of widgets that

we just created ok so now it is time to

restart our app again so to do that all

you have to do is go and fetch our out

here and this should do a hot reload and

let's go ahead and look at our app so as

you guys can see we have an app going

out here so if you scroll out here you

can find a bunch of words that get added

and this is basically our first app that

we have created and this implemented a

lot of things so first of all we learned

how to create a basic app using the

flutter terminal then we learned how we

can actually go ahead and use a hot

reload function which is pretty nifty in

my opinion then we also learned how to

ignored external libraries in this we

use the English word library you created

a steep food budget we create the

stateless widget and we also added a

lazy view and lazy scrolling to our

entire application

okay guys so if you guys can understand

any part of the application please leave

a comment in the comment section below

if you'd like me to actually take out

more flutter related tutorial videos

please also put that down in the comment

section below if you enjoyed actually

watching this video I hope you go ahead

and put a thumbs up for us and this

video because it represents better for

us on YouTube thank you for watching

with me and learning flutter today I'll

meet you guys in the next video goodbye

I hope you have enjoyed listening to

this video please be kind enough to like

it and you can comment any of your

doubts and queries and we will reply

them at the earliest do look out for

more videos in our playlist and

subscribe to any rekha channel to learn

more happy learning

Inglês (gerada automaticamente)