All right.

So now that we've gotten the location of the user's device ,the next step is to actually turn that location into actual weather data. And to do that we need to use online services where they actually collect weather data, such as Open Weather Map.

So wouldn't it be nice if we could grab the current location of the user and then tell them what the weather for their location is? Now in order to do this,we need to use what's called an API. And as Open Weather Map says they have over 20+ weather APIs.

So what exactly are these APIs? An API stands for an Application Programming Interface.

And if you take a look on Wikipedia, there's an even longer definition where it says that an API is a set of commands, functions, protocols and objects that programmers can use to create software or interact with an external system.

And the last point is it should provide developers with standard commands for performing common operations so they don't have to write the code from scratch.

Now this is quite a broad definition and as you think back to all the things that you've done even in this course, you can imagine that lots of it actually fit this definition.

So one type of APIs are APIs that are used to create software, for example Flutter.

If we take a look at the Flutter documentation, you can see that it tells you

'This is the Flutter API reference documentation.'

So by using the Flutter SDK or the libraries that are bundled together under the Flutter flag, we're essentially using an API where they've created a bunch of command functions, protocols, objects, that we're using to create software right?

We're using an API in that sense.

But there's also other types of APIs. There's APIs that allow you to interact with an external system, and an example of this is the Tinder app, which is a matchmaking app.

And when you take a look at the interface, you can see there's a section on shared friends and shared interests.

Well where did these come from?

This data comes when Tinder makes a request to Facebook asking it to get some information on a particular user saying 'This person Angela Yu, who are their friends?'

And once they get those responses back, it can check against its own users and sees out of all of those friends,which of those friends are also on Tinder? Now when we're interacting with external systems, it's a bit like being a citizen of a small town right?

There's certain buildings or certain structures that you interact with in this way.

For example you go to Dunkin' Donuts and you could make a request for some donuts, pass over some money and in return you get back your donuts as a response.

But there's also other types of APIs where you can't just go in and demand something.

For example at the bank,right?

You can't just walk in and say I want £100.

Not everybody can make that request to the bank.

So the way that it would work with the bank is that not only do you have to make a request to get some money, you also have to pass over a key saying that this is the key to unlock my account.

And then if that checks out, then and only then does the bank actually open the safe for you.

So when we're interacting with external services, we're usually trying to get a little bit of data from them. And with certain APIs, they have to make sure that you don't heat up their server so much that you're actually costing them a lot of resources. Because if you imagine an external server just as computers which are running on somebody else's electricity bill, you can imagine that if you make lots and lots of requests to that computer then you're actually costing the owner quite a lot of money.

So instead of simply having a bank where you just have some sort of notebook and everybody comes in and you say 'My name was Angela Yu. On the 25th of November I took out £100.' But actually you go in there and you just take a lot more.

Well this is where API is start demanding a key because that key is similar to having a bank teller where it asks you, 'Can I help you?', and you say, 'I'd like to withdraw £100.' And then it asks you, 'what's your password?'

So here's my passwor,d here's my API key.

And if that matches with an account that they have at the bank, then they'll say, 'Well here you go.

Here's your £100 pounds that you demanded.'

So in this case, that bank teller, that interface between you and the safe, that is the API. And it enforces certain rules that you have to abide by such as what you can request, what you can do and asking you for a password and confirming it.

That essentially is our application programming interface. Now with API you can do more than just get data.

You can also put data into their database.

So in the bank situation you might want to deposit £100.

And it's again that API that's going to handle this and take your data, verify it, validate it, make sure it goes the right place and then puts it into their own database.

You can't interact with another web server's database directly.

So I can't, for example, tell Open Weather Map that, 'you know what? I've made up some weather data.

So here you go.

Please take it and tell all of the rest of the world that actually in London right now, it's 35 degrees.'

So essentially when your app is interacting with somebody else's program or somebody else's server or somebody else's code,this API is that interface in the middle that dictates how you should interact And it's the rules of engagement

essentially. There are certain things that you can request for and if that's a valid request, then they can give you a valid response.

Now if you just went up to a bank teller and asked her, 'what's the 31st element of the periodic table?'

Because, you know, it's in your homework and you want a quick and easy way of doing it.

Well she doesn't know.

And they will return an error or in webspeak, a 404 status code which means did not find the resource that you were looking for.

Essentially you made a invalid request.

There's only certain things that they can handle and there's only certain requests that they will respond to.

So it's exactly the same with our API.

Coming back to our original definition of an API, when we try to get some data from an external system we have to do that through a series of commands and functions that have been predefined for us to use.

And it's only through these authorized ways of interacting with their system can we actually get the data that we need. And the way that we can learn about these functions or methods that they have created is by using the documentation of the API.

So for example in this case, we have the open weather map API. And in here they have detailed documentation for each of the pieces of data that we can fetch from them.

So if we wanted to access the current weather data for any location on Earth, then we can call it by using something that looks a bit like a normal URL right?

So the location of the data is api.openweathermap.org/data and then they have some sort of versioning code for which version of the API we're interacting with.

And then it says what you want to interact with and then you provide a query parameter.

So in this case you're saying, 'I want the data for the weather in London' and we can either make this request through our browser by simply putting it into the URL box at the top and we get back a series of data. So it tells us that at the moment in London, there's some drizzle and the temperature is 280K.

So in the next lesson, we're going to be looking at how we can fetch data from Open Weather Map servers through interacting with their APIs and we're going to be using a networking library from Flutter to be able to achieve this.

So for all of that and more, I see on the next lesson.