


IIT Madras
ONLINE DEGREE

Modern Application Development – I
Professor Nitin Chandrachoodan
Department of Electrical Engineering
Indian Institute of Technology, Madras
OpenAPI

Hello everyone, and welcome to this course on Modern Application Development. We have been talking about API's and seen some examples of API's so far. So, one thing that we would like to understand is, is there any way of sort of formalizing this, meaning that is there a standardized way of specifying API's, that would make it easier for someone else who is new to the API to understand what it is that you are trying to do, or even more usefully, is there some way that I can get a machine to do part of my work for me.


(Refer Slide Time: 00:46)

 IIT Madras
BSc Degree

APIs of interest for web apps

- Purpose: information hiding - neither server nor client should know details of implementation on other side
- Unbreakable contract: should not change - standardized
 - Versions may update with breaking changes

◦



Now, API's that are of interest for web apps, as you saw, with the examples of Wikipedia and CoWin and so on, they are well they are trying to do a certain form of information hiding, when I say information hiding, it is not that they are trying to hide the contents of the information from the person, what they are trying to hide is how the information is stored.

So the weather, Wikipedia uses like MySQL database or PostgreSQL or anything else at their back end, versus some other completely new proprietary mechanism of their own, should not be the end users concern. And as long as they have the API, they really do not know, and they should not have to care.


So, this information hiding is something that allows a client to use the server without knowing how the server is actually functioning. The server can also send back information to the client without bothering about whether it is a mobile device, or whether it is a desktop or what it is, it just gives the information as requested.

So, separation of concerns, in some sense, is a form of information hiding as well. And that is important for the development of web apps so that they do not break when something changes on one side or the other. And one important thing to keep in mind while thinking about this is that this is meant to be some kind of an unbreakable contract, that sounds very fancy, but what I mean is that the API should not change too easily.

And, in particular, let us say that you have built an app around CoWin which allows people to sort of log in, check various information, and so on, and the CoWin API changes, that means that your app stops working, you had nothing to do with it, you did not have control over the server and now suddenly, you are stuck, let us say you have built up a whole set of scripts to manage remote machines on the Google Cloud, you need to have something which allows you to bring up a machine programmatically, I mean you just trigger an API, which will automatically create a new VM, you need it, because the kind of load that you are seeing is suddenly scaling up very, to very large numbers.



Google has changed their API, your scripts break, there is nothing you can do, so that should not happen. So, that is why an API is expected to be standardized, that is why the version numbers are used, to sort of say that if I do need to break something, I will at least change the version number, I might retire the old version and tell people look, you have some, a few weeks or days or whatever it is within which to change, but at least you need to make it very clear that it is not the old scripts with the old version number should not be expected to work. So, those are important things to keep in mind when you are designing an API.

(Refer Slide Time: 03:36)



Documentation?

- Highly subjective - some programmers better than others at documenting
- Incomplete - what one programmer finds important may not match others
- Outdated
- Human language specific



But then comes the question of documentation, how do I document such an API? And the problem with documentation is that it is highly subjective, some programmers are very good at documentation, some are very poor documentation and some think documentation is a waste of time. Most people think documentation is a waste of time at some time or the other, that is not the way in, so do I, I do not document everything that I write.

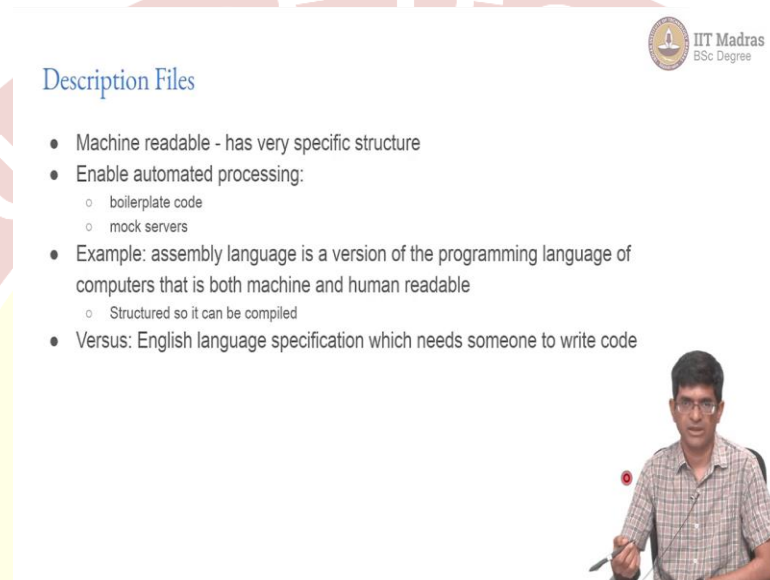
But on the other hand, it depends if you are trying to build something which is going to be used by anyone else, not just you, you need to have at least a minimum amount of documentation, or at least your code by itself should be clean enough to be able to understood. Now, documentation could also be incomplete.

Meaning that, one programmer the person who writing the code might find that, this is all self-evident, there is nothing really to document I cannot really add any useful comments over here, but the next person may wonder why is that parameter there, why is this variable being used in this fashion, it may not be obvious to another person looking at it.

It may be outdated, which is really, unfortunately common situation and the other issue, of course, is the documentation in general is human language specific, so whatever documentation you might have in one particular language, let us say you are a Chinese speaker who does not have a good command on English, you are going to struggle with English language documentation.

Can you translate it? Sure, somebody could do the translation, now you are sort of potentially opening up more problems, because the translation may not be perfect, so documentation, while necessary, it would be good to see if there are other ways by which I could sort of have a more fool proof way of describing an API.

(Refer Slide Time: 05:30)



Description Files

- Machine readable - has very specific structure
- Enable automated processing:
 - boilerplate code
 - mock servers
- Example: assembly language is a version of the programming language of computers that is both machine and human readable
 - Structured so it can be compiled
- Versus: English language specification which needs someone to write code

IIT Madras
BSc Degree

And one way of doing that is by means of something called description files. And what we are saying here is, we are going to come up with some kind of a description, which is a machine readable description, it is structured, it has a very specific sort of format and when you just think about those terms that I have been saying like structured specific format, and so on, certain kinds of mark-up languages, XML for example, sounds just about perfect, that is exactly what the language was designed for, to provide some kind of structure, which basically says, I can have I will have these tags, I will have the corresponding closing tags within that I will have some other tags, there are restrictions on what tags can be present in what places, this tag has a specific kind of meaning, all of that information can be encoded nicely in mark-up language is like XML.

So, sure XML can be it is something to keep in mind moving forward. Now, not just that a description file enables automated processing, meaning that if I have some kind of a mark-up, like that some description file, it may be possible to write a script which will just read that description and generate some boilerplate code for me, it can actually generate what is called scaffolding, some standard parts of the script, some minimal Python code, or HTML, or a


combination of HTML, Python, or maybe PHP, or whatever other language you are using, which says, look I am already creating all of these functions for you, now just put the appropriate data inside the code inside the functions.

Code your logic in there, so that it has the right things, but for example, let us say that you are doing a CRUD API, it will automatically create functions for index, show, edit, delete, and so on. And all that you need to do is just make sure that your code goes in there and you address the appropriate model for changes. So, an example of a description file or a description language is, we are used to these, we use them all the time, any programming language, even something like assembly language, is a version of the underlying programming language of a processor.

The actual programming language of the processor is those instructions, which are binary code, assembly language is basically a human readable form, but it is not just that it is human readable, it is also machine readable because it follows a very specific syntax. Similarly, a C program is something that a human can read and understand, but it is also machine readable.

What we are talking about is not something that specific, it need not be a programming language as such, it just has to be structured enough that it can be read in by a machine and processed appropriately. In particular, just an English language specification is insufficient, because it still needs somebody to read that, understand it and write code for it.

(Refer Slide Time: 08:28)



OpenAPI Specification (OAS)

- Vendor-neutral format for HTTP-based remote API specification
- Does not aim to describe all possible APIs
- Efficiently describe the common use cases
- Originally developed as Swagger - evolved from Swagger 2.0

Current version: OAS3 - v3.1.0 as of Aug 2021



OpenAPI Specification (OAS)

- **Vendor-neutral** format for **HTTP-based remote API** specification
- Does not aim to describe all possible APIs
- Efficiently describe the common use cases
- Originally developed as Swagger - evolved from Swagger 2.0

Current version: OAS3 - v3.1.0 as of Aug 2021



OpenAPI Specification (OAS)

- **Vendor-neutral** format for **HTTP-based remote API** specification
- Does not aim to describe all possible APIs
- Efficiently describe the common use cases
- Originally developed as Swagger - evolved from Swagger 2.0

Current version: OAS3 - v3.1.0 as of Aug 2021



OpenAPI Specification (OAS)

- **Vendor-neutral** format for **HTTP-based remote API** specification
- Does not aim to describe all possible APIs
- Efficiently describe the common use cases
- Originally developed as Swagger - evolved from Swagger 2.0

Current version: OAS3 - v3.1.0 as of Aug 2021



OpenAPI Specification (OAS)

- **Vendor-neutral** format for **HTTP-based remote API** specification
- Does not aim to describe all possible APIs
- Efficiently describe the common use cases
- Originally developed as Swagger - evolved from Swagger 2.0

Current version: OAS3 - v3.1.0 as of Aug 2021



So, with all of this in mind, there is something called the open API specification OAS, and this came about as something which is a vendor neutral format for HTTP based, remote API specification. So, let us break that up. We are talking about a specification, some way of describing remote API's, something that happens over the web, well remotely between a client and a server.

It is also HTTP based, which means that it is sort of oriented primarily towards the web. And it is vendor neutral, this is usually important for the simple reason that you do not want to get you know you generally we do not trust companies beyond a point, every company the bottom line is

they need to have they need their product to succeed, so there is always the chance that at some point they might say look, we need to make a change which makes it better for our servers.

So, rather than having that can we have a vendor neutral format, that is why standards and specifications standards exist, which are not controlled by any one company or organization, or rather there might be an organization but the organization has to be seen to be neutral in some way. Now, this OAS they have a nice very nicely descriptive web page that sort of gives the background information about why the specification came up, what is their motivation, what are like examples that you can use to understand and so on.

One thing they make very clear is they do not aim to describe all possible API's. And that is important because you are not trying to, for example, describe Java API's or C API's, you are restricting yourself to HTTP based remote APIs. And they decided that let us focus on describing the common used cases for a web type application rather than everything that can be done for any kind of remote procedure call.

This evolved from something called swagger, which was developed by a company called Smart bear somewhere around 2015 or so. And the current OAS standard evolved from swagger 2.0. And as of August 2021, we are at OAS 3, so OAS is the open API specification, OAS 3 is what is currently used as of August 2021.

And you are likely to see this mentioned in various places as swagger documentation. So, swagger simply because it originated from there and it is still very closely tied to that. So, there are things like swagger UI, and various swagger tools and so on, which work with open API to a large extent. And they are very useful tools, and it is probably a good idea for you to understand how they work in different contexts.