Contents

udemy
RESTful APIs

Introduction

udemy
RESTful APIs

What are Restfull APIs

API stands for Application Programming Intercace.

Restfull stands for Representational State Transfer - which means that the machines communicating do not have to 'remember' the state/condition of previous calls.

Basically one system communicating with another.

Originally you had an SDK and you interfaced your code with a library. You had to recompile for each new interface - and you were tied to a particular system or programming language.

Restfull API are web based - they can be coded in one of many languages - the sender and receiver can be coded differently - the only stipulation is that they support the same interface. This is like a contract.

Many companies have created lots of programs and features and made their functionality available by making public their APIs. Some examples are Google. A Restfull API means that it incorporates all the required information for the contracted interface.

The supported interface is a contract - it defines what it expects to be sent and what it expects to receive.

To machine A communicates with B - B could then communicate with C - but A does not have to know anything about C - it just has a contract with B.

This lets you split B up or combine B,C, D without affecting A.

Planning

udemy
RESTful APIs

Planning an API

The interface must be structured and usually follows the CRUD functionality (Create <POST>, Read <GET>, Update <PUT>, Delete <DELETE>).

First thing about the actors or objects.

Then think about the relationships (parent, child, mandatory etc)

Then think about the attributes of each object.

Once you have this - you can then model it - usually in JSON objects (JavaScript Object Notation).

Determine roles and restrictions - do you want everyone to have access to all features?

Consider using HATEOAS (Hypermediate as The Engine Of Application State) - basically this means include links to each feature as part of the returning call. A link to itself is usually required (to refresh the page).

Handle Errors by using the standard HTTP Codes.

Plan storage, do you want content always on the server or in each call or on the client - or via links?

The usual order of building an API is....

Plan - bring together actors/relationships/attributes

Design - compile a design document which will form the contract

Mock - use a tool to quickly build the interface with mocked / stubbed calls.

Iteratively build - slowly replace mocked calls with actual interfaces.

This lets the consumer team (the team receiving the calls) and the producer (the team generating the calls) to start testing immediately - in an Agile environment.

References

udemy
RESTful APIs

Best Practices
(Mulesoft)

API Design
(Microsoft)
https://docs.microsoft.com/en-us/azure/architecture/microservices/api-design

Spring

udemy
RESTful APIs

Installing Spring Boot

Navigate to www.spring.io

Select the spring boot icon - at the bottom you will see - sprint initializer

Creating a sample project

Select the Spring Initializer

Select a 'maven project' with 'java' and Spring Boot '2.0.1'

Enter group - com.kcomgroup

Enter name as the artifact - myfirstrest

Enter your dependencies

Click Generate Project - this puts the files in your downloads directory