# COMS W4156 Advanced Software Engineering (ASE)

September 27, 2022

# Agenda

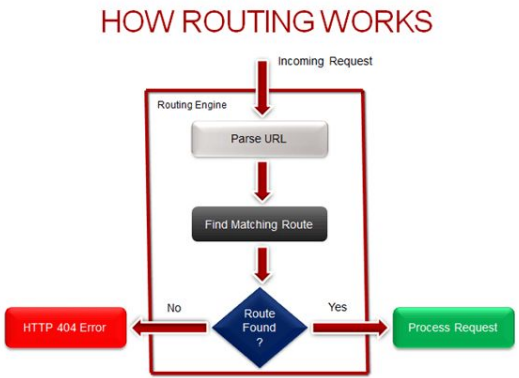
1. RESTful APIs
2. RESTful demo

# RESTful APIs

RESTful API and REST API are the same thing, “RESTful” is the technically correct term

A RESTful API is usually accessed remotely via HTTP (HyperText Transfer Protocol) or HTTPS (Secure HTTP or HTTP with Security) over the Internet with a predefined set of URLs (Uniform Resource Locators)

In this context, a URL is often called a “route”, it tells the web browser (and any intermediate internet nodes) how to get to the place on the web server where the endpoint resides

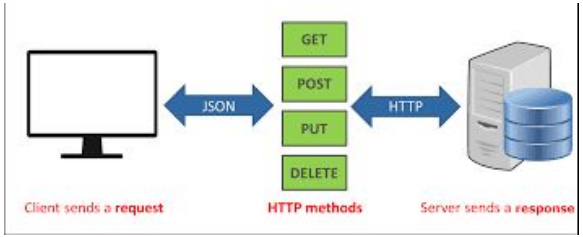



The “endpoint” is the final part of the URL after the domain and path, http://example.com/very/long/path/to/where/is/the/endpoint

# RESTful API Methods

A RESTful API consists of a set of HTTP method calls:

HTTP METHOD endpoint1 { optional body with data }
...
HTTP METHOD endpointN { optional body with data }




where “endpointN” corresponds to the API function to call with the data parameters

RESTful API terminology uses the term “method” to refer to the HTTP operations to apply to a given endpoint, typically at least GET and POST (listed here plus new one PATCH)

# Simplest RESTful API Calls


Consider the URL http://example.com/very/long/path/to/where/is/the/endpoint

If you click a link like this one on a page in your browser, the browser will use HTTP to connect to the web server at example.com (domain) and tell it to access the /very/long/path/to/where/is/the/endpoint file directory path from the webroot

A client program that is not a browser can also use HTTP to connect to a server or service that is not a web server at example.com (domain) and tell it to handle /very/long/path/to/where/is/the/endpoint

In the web case, this typically fetches some content located at /very/long/path/to/where/is/the/endpoint, which is then returned to and rendered by your browser. The ‘rendering’ could give you text, image, audio, video, code, or any other data

But this could instead run a program denoted by /very/long/path/to/where/is/the/endpoint and then return the results of running that program, with the results (not the program) returned to the client to process

# What do RESTful API calls return?

REST = REpresentational State Transfer because accessing the resource *transfers* to the caller a *representation* of the current *state* of the resource

The HTTP caller (client) can be any program, the requestor does not need to be a web browser

And the callee can be any program, the response does not have to come from a web server

When the HTTP caller is your program,
you are using a RESTful API

When the HTTP callee is your program,
you are implementing a RESTful API

# What is a “Representation”?

The same resource (or resource state) may have many different representations (variants)

- A page of text might be presented in HTML or plain text
- The words in the text might be written in English or Chinese
- The characters in the text might be encoded in ASCII or UTF-8
- The page might be organized differently for a PC versus a mobile device

The client might specify the representation(s) it can handle by including optional fields in the HTTP header or the server might send multiple representations and the client uses those it can handle

Determining what to send/receive is called *content negotiation*

# RESTful APIs with Basic Parameters


http://example.com/path/to/resource?query

A query string is something like field1=value1&field2=value2&field3=value3

If you click a query URL like this one in your browser, the browser will use HTTP to connect to the web server at example.com, tell the program at the /path/to/resource endpoint to run, and give it the parameters value1, value2 and value3.  The browser then renders the results of running that program with the parameters

If a client program constructs a query like this one and uses HTTP to send it to the server or service at example.com/path/to/resource, with the parameters value1, value2 and value3, the server program usually runs the designated function with those parameters and returns the result to the client for processing. But the server program might do something else, such as send the result to a third party, according to the semantics of the API function

# Where does the Query come from?

Usually a query URL would not be sitting there embedded in a page (or in the client code) with the field values already filled in

Instead the values of the fields would likely be computed somehow, in the browser case possibly by showing the user a form to fill in - then the form displays the fields and the user enters the values

Again the HTTP caller client can be any program, it does not need to be a web browser, and the callee server can be any program, it does not need to be a web server

When the caller /callee is your program, you are again using/implementing a RESTful API

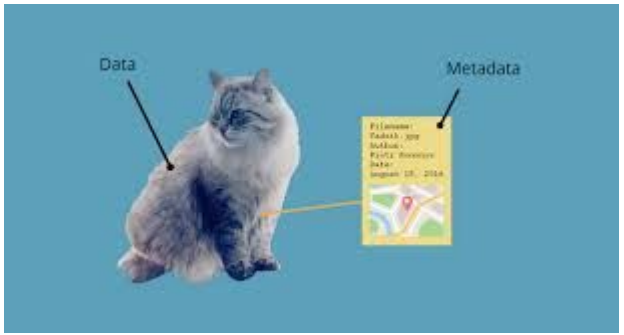
# Where are the GET and POST?

A URL like http://example.com/path/to/resource would presumably not change the state of the resource (except maybe to increment metadata like how many times the page has been accessed or update the timestamp for the most recent access)

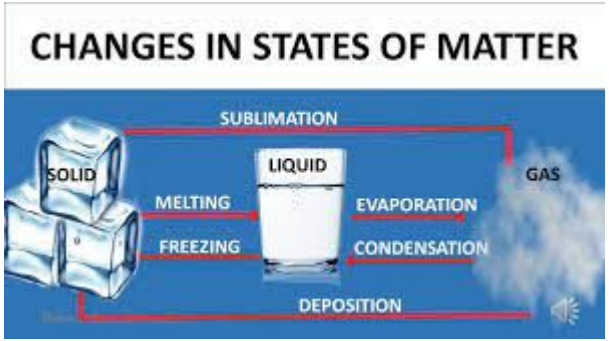
If your API call only intends to fetch something with no side-effects on the resource state, use GET

A URL query like
http://example.com/path/to/resource?field1=value1&field2=value2&field3=value3
also should not have side-effects

If your API call only intends to fetch something, e.g., the field values will be used for search with no side-effects on the resource state, then again use GET - in this case with the query fields/values included in the HTTP header



# Many Services Have State




Let's say we wanted to update some data to *set* field1=value1&field2=value2&field3=value3

That is, your API call intends to *change* the state of the resource based on the provided field values, then use POST and put the fields/values in the body of the HTTP request rather than the header

That is, http://example.com/path/to/resource goes in the http *header* and the query string field1=value1&field2=value2&field3=value3 goes in the http *body* rather than the header

# Many Services Compute State


Let's say we wanted to update some data based on
the result of running the resource program with parameters
field1=value1&field2=value2&field3=value3

Again, if the API call is intended to *change* the state of the resource based on the provided field values, then the API should expect the POST method and expect to find the fields/values in the body of the HTTP request rather than the header (and might produce an error status code otherwise)

Again http://example.com/path/to/resource goes in the http header and the query string field1=value1&field2=value2&field3=value3 goes in the http body rather than the header

The code that implements the client caller and server endpoint is just like any other code you might write, except it probably uses json, xml, or some other easily serializable format for representing data

# Agenda

1. RESTful APIs
2. RESTful demo

# RESTful demo: Peter McNeely

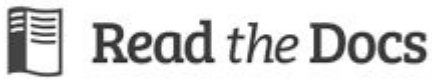
demo presentation

# Read The Docs

More information about RESTful APIs here

Example Java RESTful API frameworks: Spring, Javalin

Example C++ RESTful API frameworks: benchmark comparing six frameworks



# Your Team Projects Need to Implement an API

Your service has to implement some API, but it does not need to be a RESTful API, and could involve interacting with other services, RPC, mediation, callbacks, events, etc. - but no end-user UI, neither GUI nor CLI

An administrative console is ok, just make sure your service cannot be mistaken for an “app”.  If you find yourself describing or even thinking about your server as an “app”, you’re doing it wrong

# Upcoming Assignments

Revised Project Proposal: due Monday October 3 - meet with your Mentor to discuss your preliminary proposal *before* submitting revision


Mentors will be assigned soon™ based on preliminary proposals. It's ok to submit revised proposals late if we take too long assigning mentors

# Next Class

RPC APIs

gRPC demo

# Ask Me Anything