



HTTP + REST APIs

Tristan Wiesepape



Step 1: Inspect element

Step 2: ???

Step 3: Solving medium web challenges



How do we talk over the network

- TCP
 - Connect to hostname
 - Create a bidirectional pipe
 - Both sides can read and write bytes
 - Hard to use for applications
 - No standard format for data
 - Each app has to design its own format
 - No way to communicate in units of an entire message
 - Hard to work with non fixed size messages



How does HTTP work

- Built on TCP
- Request and Response
 - Communicate in units of entire messages instead of bytes
 - Plain text
 - Human readable
 - Contains a header and body
 - Client only sends requests, server only responds to requests



HTTP Request

- This request fetches file.html from website.com
- Method
 - GET, POST, PATCH, PUT, DELETE, OPTIONS, etc..
- Path
- Version
 - We will only discuss HTTP/1.1
- Headers
 - Arbitrary key-value store
- Body
 - Arbitrary data

```
GET /path/file.html HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: website.com
```



HTTP Response

- Response to previous request. Contains contents of file.html
- Version
 - We will only discuss HTTP/1.1
- Status
 - 200 is success
 - 400 is client error
 - 500 is server error
- Headers
 - Arbitrary key-value store
- Body
 - Arbitrary data

```
HTTP/1.0 200 OK
Content-Length: 14
Content-type: text/html
Date: Tue, 28 Sep 2021 17:46:49 GMT
Last-Modified: Tue, 28 Sep 2021 17:46:27 GMT
Server: SimpleHTTP/0.6 Python/3.9.7

file contents
```



Data Transport

- The request and response contents can be any arbitrary data
 - HTML
 - Binary data
 - Form encoded
 - JSON
 - XML
- Dont worry too much about these terms, learn them as you need them



Web servers

- Now that we have requests, how do we respond to them? What do we send back?
- Request to “example.com/a/b/c” returns the file “a/b/c”, relative to the web root
 - The Apache web root is “/var/www/html”, so the request “example.com/file” returns “/var/www/html/file”
- Might return the file contents verbatim
- Might do something with the file before returning it (PHP)



REST APIs

- Design philosophy, not a technology
- Run code instead of returning files
- Used to query and edit data in a way that multiple platforms can easily deal with
- Based on HTTP methods and url paths



Example

- We are the proud owner of a widget company
- We have a database of widgets we want to query and edit
- Multiple different apps need to use this database, so we want a single API that can be used by a wide variety of programming languages and platforms
- Our database consists of a list of widgets, and each widget has a id and a name



API Documentation

GET /widgets -> Returns a list of all widgets


GET /widgets/id -> Returns the widget with the given id

POST /widgets -> Makes a new widget

PUT /widgets/id -> Make a new widget with the given id

PATCH /widgets/id -> Update the widget with the given id

DELETE /widgets/id -> Delete the widget with the given id



```
GET /widgets HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8000
User-Agent: HTTPie/2.5.0
```

```
HTTP/1.0 200 OK
Content-Length: 161
Content-type: application/octet-stream
Date: Tue, 28 Sep 2021 18:21:42 GMT
Last-Modified: Tue, 28 Sep 2021 18:20:57 GMT
Server: SimpleHTTP/0.6 Python/3.9.7
```

```
{
  "data": [
    {
      "id": 10,
      "name": "BOB"
    },
    {
      "id": 20,
      "name": "JOE"
    }
  ]
}
```

```
GET /widgets/10 HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8000
User-Agent: HTTPie/2.5.0
```

```
HTTP/1.0 200 OK
Content-Length: 36
Content-type: application/octet-stream
Date: Tue, 28 Sep 2021 18:23:02 GMT
Last-Modified: Tue, 28 Sep 2021 18:22:45 GMT
Server: SimpleHTTP/0.6 Python/3.9.7
```

```
{
  "id": 10,
  "name": "BOB"
}
```



API Design

- The HTTP Method tells you what to do, the URL tells you what to do it to
- The correlation between URL paths and data is completely arbitrary
 - We can have each thing in our database have its own set of endpoints, or have one set of endpoints edit multiple things in our database
- We can have endpoints that generate the returned values on demand (i.e. we don't query them from a database)
- We can really do whatever we want



REST “Defaults”

- Things are often represented in JSON
- Sometimes all api urls will be preceded with a version
 - I.e. all requests go to “example.com/api/v1/...”
- Each thing has its own set of endpoints
- Endpoints are paginated
 - Big responses are cut up into chunks
 - Instead of “GET example.com/widgets”
 - “GET example.com/widgets?page=1”, then “GET example.com/widgets?page=2”



Tools

- Browsers
 - Hard to control and script
 - Burp suite will let you see all of the requests made by your browser
- Programming Languages
 - Almost every programming language has some way to send HTTP requests
 - Python has requests, Javascript has fetch, Java has `java.net.http`, etc
- Command Line Tools
 - curl
 - Extremely popular tool
 - wget
 - httpie
 - My favorite
 - All of them do they same thing, they just have different syntax



Security

- The first step to exploiting a system is understanding how to communicate with it
- CTF
 - Scripting HTTP requests can automate manual work
 - Editing different parts of your request can be used to solve challenges
- Software Development
 - REST APIs are the bridge between frontend and backend development
 - Knowing how HTTP works is super useful for debugging
- Application Security
 - A surprising number of vulnerabilities come from sending HTTP requests the developers didn't expect
 - Number 1 on the OWASP top 10 is Broken Access Control