

ReactJS Networking

Isumi
Batam, Jan 2020

GOALS

1. Http Method
2. Postman
3. Axios

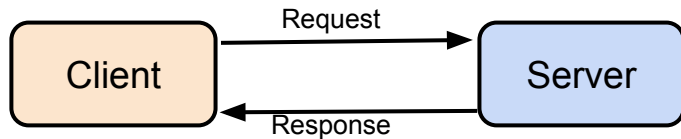


1. Http Method

Before we start learn about HTTP Method, let's warm up by understanding more about ***client*** and ***server***.



Client & Server



Client is a computer or software that accesses a service made available by a server, like a mobile app (Facebook, Instagram, etc), browsers, desktop app, etc.

Server is a computer or software that provide services in a centralized resources, it stores data or process all request and provide response to client.

Request is when a client access available service on server.

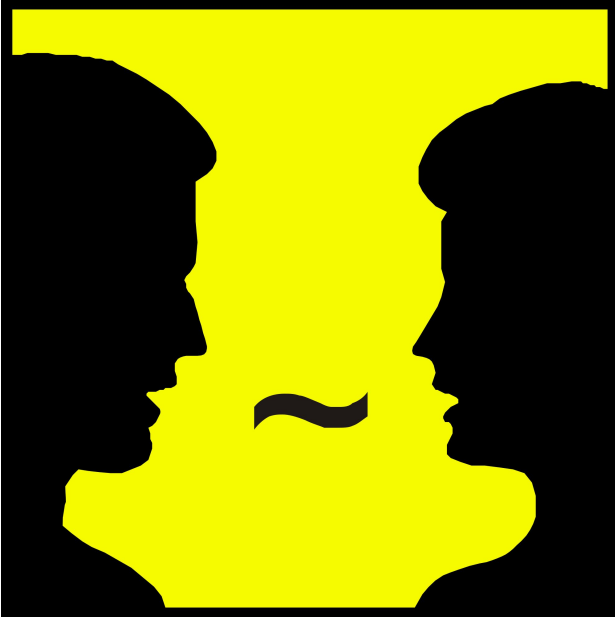
Response is when a server send data back to client based on request from a client.

But how do the request work?

What's happen behind the request or response?



Protocols



A set of rules that client and server agree to use when communicating



Protocols

HTTP/S

FTP

SMTP

Many more...

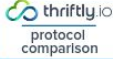
There are various protocols available:
HTTP (Hypertext Transfer Protocol),
FTP (File Transfer Protocol),
SMTP (Simple Mail Transfer Protocol) and
many more.

But in our case we will use HTTP / HTTPS
because it was developed to facilitate [hypertext](#)
and the *Word Wide Web*.

HTTP / FTP / SMTP is like the English
language, a way that client & server can
communicate each other.



API Protocols

	First released	Formatting type	Key strength
SOAP	Late 1990s	XML	Widely used and established
REST	2000	JSON, XML, and others	Flexible data formatting
JSON-RPC	mid-2000s	JSON	Simplicity of implementation
gRPC	2015	Protocol buffers by default; can be used with JSON & others also	Ability to define any type of function
GraphQL	2015	JSON	Flexible data structuring
Thrift	2007	JSON or Binary	Adaptable to many use cases

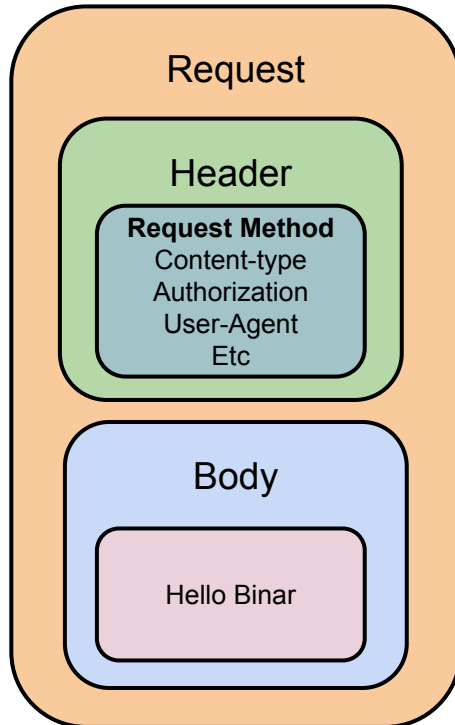


Request and response

A request sent by clients and response sent by server via http basically contains message data consist of header and body.



Request

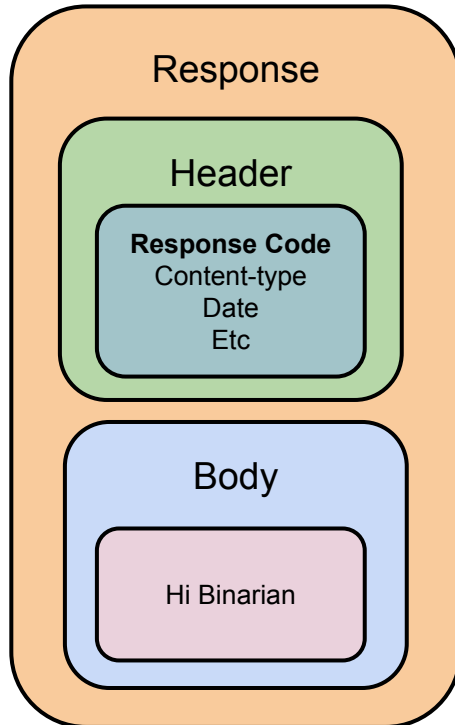


When a client send request to a service via browser, it will automatically attach client's information in the header **after the request method** (*GET*, *POST*, *UPDATE*, *DELETE*, etc), for example the browser's name, version, type of messages, etc.

Alongside with header there's body which contains the message we want to send to server, for example a plain text `Hello Binar`.



Response



The main difference between `request` and `response` header is a request need a *Request Method*, **but a response need a Response Code** (200, 301, 503, etc) so the client can handle response based on the code. Like a request, a response can contains addition header messages too.



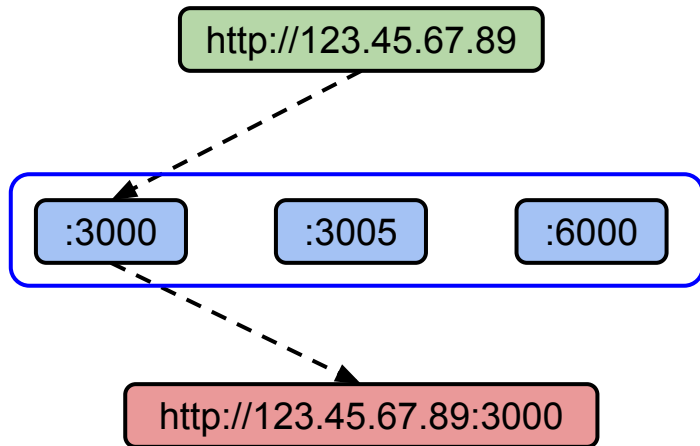
IP Address & Ports

Every school have address so we know where we should be going when we want to study, also it has class rooms for specific study so we know what room that has course we take.

Similar concept is used in IP Address & Ports, IP Address used to point an address and ports used to point the services.



IP Address & Ports



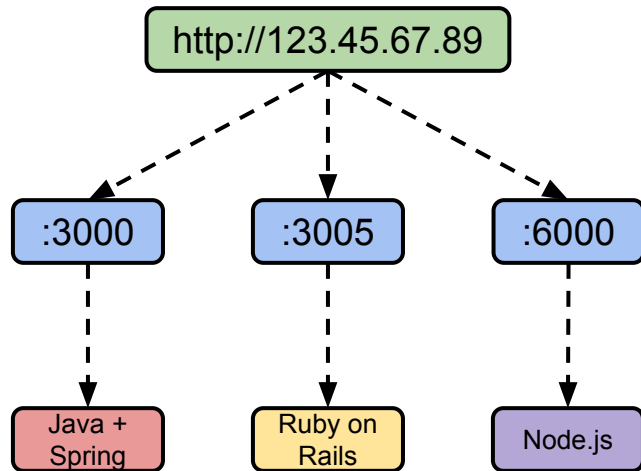
IP Address has 2 versions, *IPv4* and *IPv6*. Basically they are same, the main difference is the size (IPv4 32bits and IPv6 128 bits). But for now let's focus on IPv4.

An IPv4 Address has size 32bits, you can see [here](#) for the detail, it consist of 4 decimal numbers separated by dots, each ranging from 0 to 255.

Ports has range from 0 to 65535, it is written after IP Address separated by colon (:), for example <http://123.45.67.89:3000>.



IP Address & Ports



We already use IP Address, but why we need ports?

Like a classrooms that provide various study, ports are used to provide various services for clients or server itself. We can use ports from 0 - 65535, but there are [well-known ports](#) that maybe already used by OS, so we can't use those ports.

In *microservices architecture*, each port can be used to provide specific services or even application, auth service is on port 3000 developed with Java, order service is on port 6000 developed with Node.js, etc.



Explore Client and Server

1. Client & Server : <https://simple.wikipedia.org/wiki/Client-server>
2. Client & Server (Video) :
<https://www.youtube.com/watch?v=qSAze9b0wrY&index=11&list=PL4cUxeGkcC9gcy9IrvMJ75z9maRw4byYp>
3. Request : <https://www.w3.org/Protocols/rfc2616/rfc2616-sec5.html>
4. Response : <https://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html#sec6>

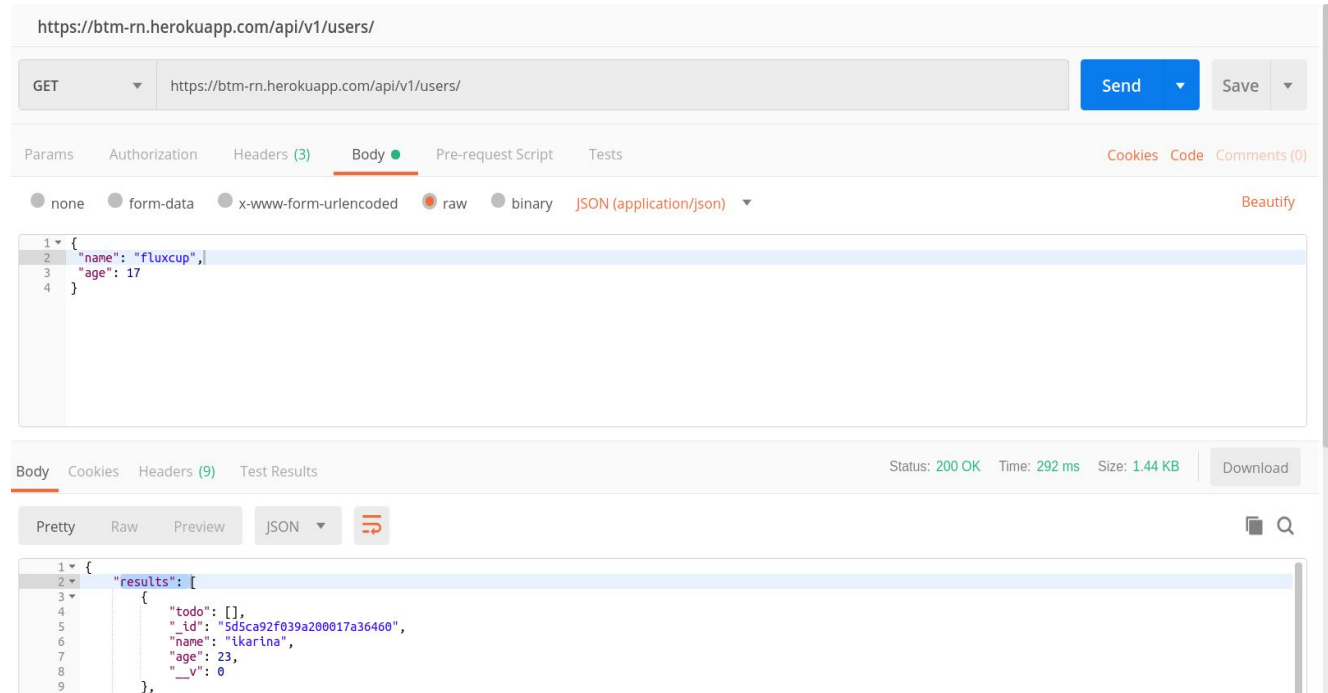


2. Postman

Tool for API testing.

Notice:

API (Application Programming Interface) which allows software applications to communicate with each other via API calls.



Tutorial: read this [article](#)



3. Axios

Promise-based HTTP library for performing HTTP requests on both Nodejs and Browser.

1. Install:

```
npm install axios -S
```

2. Import axios library at your component file:

```
import axios from 'axios';
```

3. Insert axios method inside lifecycle component:

```
componentDidMount() {}
```

Notice:

Explore others HTTP Request Libraries: Superagent, Request, Fetch, Supertest

Exercise

Do mapping the data using lifecycle component and axios. ;)

Used this endpoint:

<https://btm-rn.herokuapp.com/api/v1/users/>