

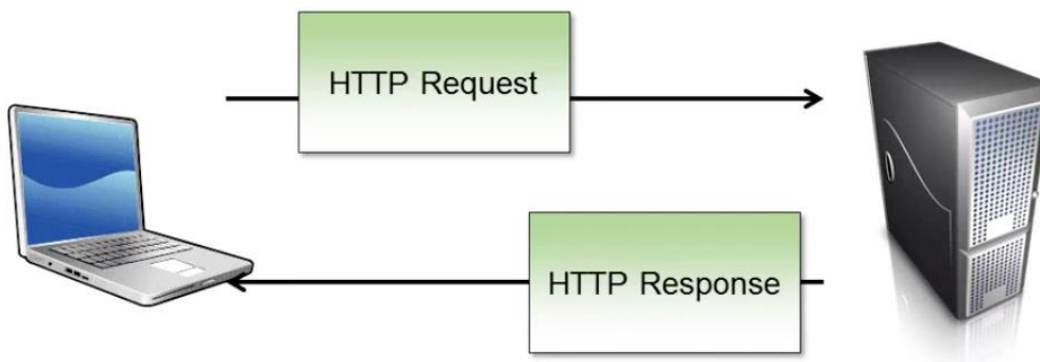
# FSW-115: Communication Over the Web – HTTP, AJAX and APIs

## Week 1: HTTP & JSON

## Week 2: URLs & GET Request with Postman

- Hypertext Transfer Protocol (HTTP):
  - technology on how data is exchanged over the web.
  - a protocol is a predefined set of rules that allows computers to , communicate with each other (tcp, ip, http, ftp, ssh, smtp, etc).
  - clients (browser) transmit message containing http methods (verbs) that tell their **request** to the receiving computer (server).
  - server receives these requests, reacts and then send the **response** back to the client.

**A single HTTP Transaction (request – response cycle)  
(always initiated by the client):**



<https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>

HTTP Request Methods: (tells the server what you want to do):

- GET – Retrieve a resource (requesting data)
- POST – Create a new resource (sending data)
- PUT – Update a resource (update data)
- DELETE – Remove a resource (remove data)

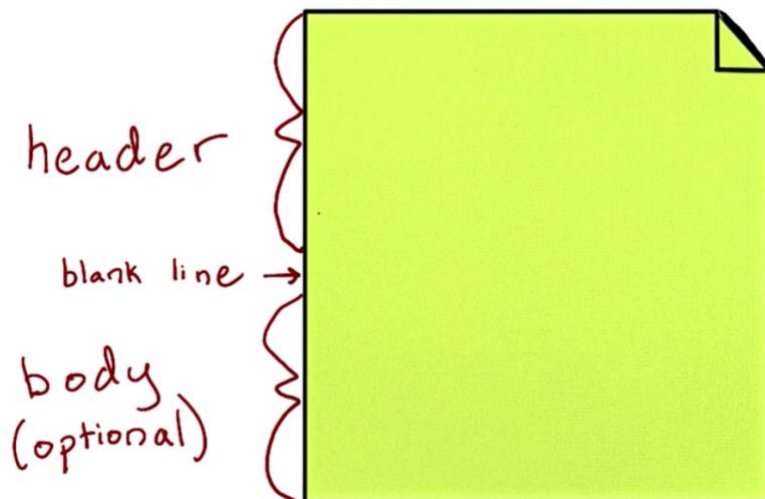
What is an API and a REST API?

- APIs connect one application to the data services of another application. It is mainly just a set of functions.
- REST (Representation State Transfer) is an architectural style for networked applications on the web. It is a guideline on how to build an API.
- REST APIs are web service APIs which uses URIs and HTTP and JSON for its data format

Great video: <https://www.youtube.com/watch?v=SLwpgD8n3d0>

What is a Request?

- A way to exchange data between a client and server
- Client send the request
- Server responds to the request
- Each **request** is sent to a specific URL
- Message returned by the server is called the **response**



```
GET puppies.html HTTP/1.1
Host: www.puppyshester.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35
```

**puppyId=12345&name=Fido+Simpson**

#### Request and Response Data

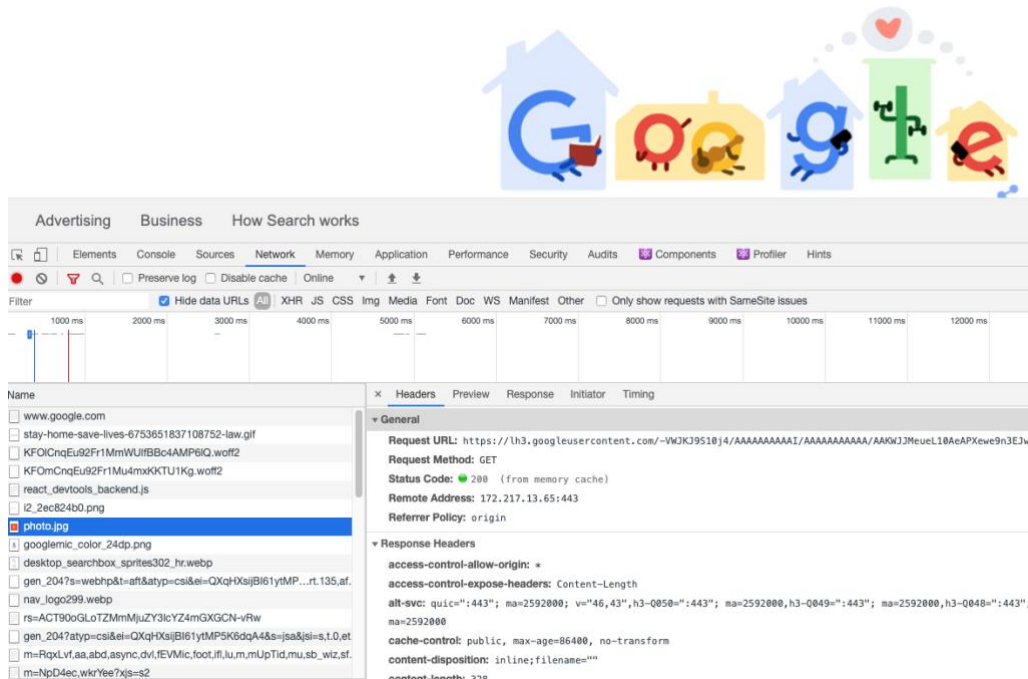
- Form data
- JSON ({"key":"value"})
- Files (images, .zip, .html, .css)
- XML (structured data, but used less now that JSON is more popular)

#### Request Components

- Method (GET, POST, DELETE, PUT)
- Data (optional)
- Headers (meta information – type of data, etc.)
- Authentication (token – verify who you are; usually stored in header)

#### Response Components

- Data (type of data you are requesting - JSON, .html, .css, images, etc.)
- Status Code (2xx, 3xx, 4xx, 5xx); 1XX – Information, 2XX – Success, 3XX – Redirect, 4XX – Client Error, 5XX – Server Error
- Headers (tells you the type of data returned)



A web page can contain many HTTP requests. (network tab).

Status Codes (issued by the server from a request)

## HTTP Status Codes

This page is created from HTTP status code information found at [ietf.org](http://ietf.org) and [Wikipedia](http://Wikipedia). Click on the **category heading** or the **status code** link to read more.

### 1xx Informational

100 Continue

101 Switching Protocols

102 Processing (WebDAV)

### 2xx Success

★ 200 OK  
203 Non-Authoritative Information  
206 Partial Content  
226 IM Used

★ 201 Created  
★ 204 No Content  
207 Multi-Status (WebDAV)

202 Accepted  
205 Reset Content  
208 Already Reported (WebDAV)

### 3xx Redirection

300 Multiple Choices  
303 See Other  
306 (Unused)

301 Moved Permanently  
★ 304 Not Modified  
307 Temporary Redirect

302 Found  
305 Use Proxy  
308 Permanent Redirect (experimental)

### 4xx Client Error

★ 400 Bad Request  
★ 403 Forbidden  
406 Not Acceptable  
★ 409 Conflict  
412 Precondition Failed  
415 Unsupported Media Type  
418 I'm a teapot (RFC 2324)  
423 Locked (WebDAV)  
426 Upgrade Required  
431 Request Header Fields Too Large  
450 Blocked by Windows Parental Controls (Microsoft)

★ 401 Unauthorized  
★ 404 Not Found  
407 Proxy Authentication Required  
410 Gone  
413 Request Entity Too Large  
416 Requested Range Not Satisfiable  
420 Enhance Your Calm (Twitter)  
424 Failed Dependency (WebDAV)  
428 Precondition Required  
444 No Response (Nginx)  
451 Unavailable For Legal Reasons

402 Payment Required  
405 Method Not Allowed  
408 Request Timeout  
411 Length Required  
414 Request-URI Too Long  
417 Expectation Failed  
422 Unprocessable Entity (WebDAV)  
425 Reserved for WebDAV  
429 Too Many Requests  
449 Retry With (Microsoft)  
499 Client Closed Request (Nginx)

### 5xx Server Error

★ 500 Internal Server Error  
503 Service Unavailable  
506 Variant Also Negotiates (Experimental)  
509 Bandwidth Limit Exceeded (Apache)  
598 Network read timeout error

501 Not Implemented  
504 Gateway Timeout  
507 Insufficient Storage (WebDAV)  
510 Not Extended  
599 Network connect timeout error

502 Bad Gateway  
505 HTTP Version Not Supported  
508 Loop Detected (WebDAV)  
511 Network Authentication Required

<https://restfulapi.net/http-status-codes/>

## Data Format: JSON (JavaScript Object Notation)

[https://www.w3schools.com/whatis/whatis\\_json.asp](https://www.w3schools.com/whatis/whatis_json.asp)

Example:

<https://codepen.io/frankstepanski/pen/MWwdzqO>

<https://codepen.io/frankstepanski/pen/JjdQYGp>

Validate and generators:

<https://jshint.com/>

<https://jsonlint.com/>

<https://jsonformatter.org/>

<https://www.json-generator.com/>

Video: <https://www.youtube.com/watch?v=iiADhChRriM>

## JSON

```
{  key    value
  "name": "Andrew",
  "employer": "LinkedIn",
  "hobbies": "ping-pong"
}
```

## Postman

The screenshot shows the Postman application interface. At the top, there's a toolbar with buttons for 'New', 'Import', 'Runner', and 'My Workspace'. Below this, a tab bar shows 'GET http://swapi...' as the active request. The main area displays the request details: 'GET' method and 'http://swapi.co/api/people/2' URL. Below the request bar, there are tabs for 'Params', 'Authorization', 'Headers (9)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Params' tab is selected, showing a table with 'KEY' and 'VALUE' columns. Below the table, there's a 'Query Params' section. At the bottom, the 'Body' tab is selected, showing the response in 'JSON' format. The response is a JSON object representing C-3PO, with fields like 'name', 'height', 'mass', 'hair\_color', 'skin\_color', 'eye\_color', 'birth\_year', 'gender', 'homeworld', and 'films'.

```
1 {
2   "name": "C-3PO",
3   "height": "167",
4   "mass": "75",
5   "hair_color": "n/a",
6   "skin_color": "gold",
7   "eye_color": "yellow",
8   "birth_year": "112BBY",
9   "gender": "n/a",
10  "homeworld": "https://swapi.co/api/planets/1",
11  "films": [
12    "https://swapi.co/api/films/2/",
13    "https://swapi.co/api/films/5/",
14    "https://swapi.co/api/films/4/",
15    "https://swapi.co/api/films/6/"
16  ]
17 }
```

Learning: <https://learning.postman.com/getting-started/>

Running through the steps in this article: <https://coursework.vschool.io/postman/>

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettingsCookiesCode

▼ Headers (1)

	KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets ▼
<input type="checkbox"/>	Accept	application/vnd.github.mercy-preview+json				
	Key	Value	Description			

▼ Temporary Headers (8) ⓘ

	KEY	VALUE
	User-Agent	PostmanRuntime/7.22.0
	Accept	*/*
	Cache-Control	no-cache
	Postman-Token	3e06cd58-ff5a-48c9-bc28-bca807cbfd68
	Accept-Encoding	gzip, deflate, br
	Referer	https://swapi.co/api/people/2
	Cookie	__cfduid=d1bd27966fdb96d36766a9799bd418b1a1585958984
	Connection	keep-alive

GET▼https://api.vschool.io/frankstepanski/todo/

Send▼Save▼

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettingsCookiesCode

☐ none☐ form-data☐ x-www-form-urlencoded☒ raw☐ binary☐ GraphQL

JSON▼

Beautify

1

BodyCookiesHeaders (8)Test Results

Status: 200 OKTime: 26msSize: 275 BSave Response▼

PrettyRawPreviewVisualize

JSON▼

1

[ ]

GET Untitled Re...

Launchpad

GET express de...

Bootcamp

POST https://api...

+

...

No Environment

Untitled Request

Comments (0)

POST

https://api.vschool.io/frank/todo/

Send

Save

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Code

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

1

{

2

"title": "Prepare live classroom presentation."

3

}

Body

Cookies

Headers (8)

Test Results

Status: 200 OK

Time: 53ms

Size: 403 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

1

{

2

"completed": false,

3

"\_id": "5e87d267f468d266b6c890f2",

4

"title": "Prepare live classroom presentation.",

5

"sessionId": "frank",

6

"\_\_v": 0

7

}

POST

https://api.vschool.io/frankstepanski/todo/

Send

Save

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Code

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

1

{

2

"title": "Pay bills online"

3

}

Body

Cookies

Headers (8)

Test Results

Status: 200 OK

Time: 23ms

Size: 393 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

1

{

2

"completed": false,

3

"\_id": "5e87d31cf468d266b6c890f7",

4

"title": "Do work for class",

5

"sessionId": "frankstepanski",

6

"\_\_v": 0

7

}

GET

https://api.vschool.io/frankstepanski/todo/

Send

Save

Body

Cookies

Headers (8)

Test Results

Status: 200 OK

Time: 19ms

Size: 736 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1  [
2    {
3      "completed": false,
4      "_id": "5e87d2fff468d266b6c890f5",
5      "title": "Take a walk",
6      "sessionId": "frankstepanski",
7      "__v": 0
8    },
9    {
10     "completed": false,
11     "_id": "5e87d30df468d266b6c890f6",
12     "title": "Take a nap",
13     "sessionId": "frankstepanski",
14     "__v": 0
15   },
16   {
17     "completed": false,
18     "_id": "5e87d31cf468d266b6c890f7",
19     "title": "Do work for class",
20     "sessionId": "frankstepanski",
21     "__v": 0
22   },
23   {
24     "completed": false,
25     "_id": "5e87d333f468d266b6c890f8",
26     "title": "Pay bills online",
27     "sessionId": "frankstepanski",
28     "__v": 0
29   }
30 ]
```

GET

https://api.vschool.io/frankstepanski/todo/

Body Cookies Headers (8) Test Results

Pretty

Raw

Preview

Visualize

JSON

```
1  [
2    {
3      "completed": false,
4      "_id": "5e87d2fff468d266b6c890f5",
5      "title": "Take a walk",
6      "sessionId": "frankstepanski",
7      "__v": 0
```



GET https://api.vschool.io/frankstepanski/todo/5e87d2fff468d266b6c890f5 Send Save

Body Cookies Headers (8) Test Results Status: 200 OK Time: 111ms Size: 387 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "completed": false,
3   "_id": "5e87d2fff468d266b6c890f5",
4   "title": "Take a walk",
5   "sessionId": "frankstepanski",
6   "__v": 0
7 }
```

PUT https://api.vschool.io/frankstepanski/todo/5e87d2fff468d266b6c890f5 Send Save

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "title": "Take a really long walk"
3 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 114ms Size: 399 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "completed": false,
3   "_id": "5e87d2fff468d266b6c890f5",
4   "title": "Take a really long walk",
5   "sessionId": "frankstepanski",
6   "__v": 0
7 }
```

PUT https://api.vschool.io/frankstepanski/todo/5e87d2fff468d266b6c890f5 Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "title": "Take a really long walk"
3 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 111ms Size: 387 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "completed": false,
3   "_id": "5e87d2fff468d266b6c890f5",
4   "title": "Take a walk",
5   "sessionId": "frankstepanski",
6   "__v": 0
7 }
```

GET https://api.vschool.io/frankstepanski/todo/ Send Save

Body Cookies Headers (8) Test Results Status: 200 OK Time: 24ms Size: 748 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "completed": false,
4     "_id": "5e87d2fff468d266b6c890f5",
5     "title": "Take a really long walk",
6     "sessionId": "frankstepanski",
7     "__v": 0
8   },
9   {
10    "completed": false,
11    "_id": "5e87d30df468d266b6c890f6",
12    "title": "Take a nap",
13    "sessionId": "frankstepanski",
14    "__v": 0
15  },
16  {
17    "completed": false,
18    "_id": "5e87d31cf468d266b6c890f7",
19    "title": "Do work for class",
20    "sessionId": "frankstepanski",
21    "__v": 0
22  },
23  {
24    "completed": false,
25    "_id": "5e87d333f468d266b6c890f8",
26    "title": "Pay bills online",
27    "sessionId": "frankstepanski",
28    "__v": 0
29  }
30 ]
```

PUT https://api.vschool.io/frankstepanski/todo/5e87d2fff468d266b6c890f5 Send Save

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "title": "Take a very long walk",
3   "description": "Found a new park",
4   "price": "No price",
5   "imgUrl": "n/a",
6   "completed": false
7 }
```

Body Cookies Headers (8) Test Results Status: 500 Internal Server Error Time: 24ms Size: 368 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "error": "Cast to number failed for value \"No price\" at path \"price\""
3 }
```

The screenshot displays a REST client interface with two panels. The top panel shows a POST request to `https://api.vschool.io/frankstepanski/todo/5e87d2fff468d266b6c890f5` with a JSON body. The bottom panel shows the response, which is a JSON array of four todo items. The status is 200 OK, time is 30ms, and size is 804 B.

**Request:**

```
POST https://api.vschool.io/frankstepanski/todo/5e87d2fff468d266b6c890f5
Body
{
  "title": "Taking a very long walk",
  "description": "Found a new park",
  "price": 0,
  "imgUrl": "n/a",
  "completed": false
}
```

**Response:**

```
GET https://api.vschool.io/frankstepanski/todo/
Status: 200 OK Time: 30ms Size: 804 B
Body
[
  {
    "completed": false,
    "_id": "5e87d2fff468d266b6c890f5",
    "title": "Take a very long walk",
    "sessionId": "frankstepanski",
    "__v": 0,
    "description": "Found a new park",
    "imgUrl": "n/a",
    "price": 0
  },
  {
    "completed": false,
    "_id": "5e87d30df468d266b6c890f6",
    "title": "Take a nap",
    "sessionId": "frankstepanski",
    "__v": 0
  },
  {
    "completed": false,
    "_id": "5e87d31cf468d266b6c890f7",
    "title": "Do work for class",
    "sessionId": "frankstepanski",
    "__v": 0
  },
  {
    "completed": false,
    "_id": "5e87d333f468d266b6c890f8",
    "title": "Pay bills online",
    "sessionId": "frankstepanski",
    "__v": 0
  }
]
```

## JSON Parsing Errors

Normally you will not be manually creating a JSON file from scratch. JSON data will normally be created by converting a JS Object or from a database. But whether you create it from hand or programmatically, you may run into parsing errors.

Here is a great article on the most common error and how to try and fix them:

<https://philosophy.org/code/fixing-syntaxerror-unexpected-string-token-in-json-at-position/>