

REST and JSON

JSON

JavaScript Object Notation

- Goal: Transfer data
 - Between computers / processes
 - Between programming languages
- Best for "data" object.
 - Key/Value pairs. Dictionaries, Arrays
- Not great for object relationships
 - Linked Lists, Graphs, OOP

JSON vs XML

fight!

```
<?xml version="1.0" encoding="UTF-8"?>
<cas:serviceResponse xmlns:cas="http://www.yale.edu/tp/cas">
  <cas:authenticationSuccess>
    <cas:user>fisherm</cas:user>
    <cas:attributes>
      <cas:dbkey>1111111111111111</cas:dbkey>
      <cas:emplid>22222222</cas:emplid>
      <cas:activestudent>0</cas:activestudent>
      <cas:activeemployee>1</cas:activeemployee>
    </cas:attributes>
  </cas:authenticationSuccess>
</cas:serviceResponse>
```

XML

```
{
  "serviceResponse": {
    "authenticationSuccess": {
      "user": "fisherm",
      "attributes": {
        "dbkey": "1111111111111111",
        "emplid": "22222222",
        "activestudent": 0,
        "activeemployee": 1
      }
    }
  }
}
```

JSON

JSON Formal Grammar

- Whitespace is unimportant
- These are all equivalent

```
{  
  "name": "Mark",  
  "netid": "fischerM"  
}
```

```
("name": "Mark", "netid": "fischerM")
```

```
{"name": "Mark", "netid": "fischerM"}
```

```
{  
"name":  
"Mark",  
"netid":  
"fischerM"  
}
```

7

JSON Objects

- Objects are defined with curly braces
- Key/Value pairs are separated by a colon
- Keys are strings
- Values can be strings, numbers, boolean, null, objects, or arrays
- "key": "value" pairs separated by commas
- Trailing commas are not allowed

```
{  
  "name": "Mark",  
  "netid": "fischerM"  
}
```

```
{  
  "name": "Mark",  
  "netid": "fischerM",  
}
```

8

JSON Objects

- Keys must be strings
- Double Quotes are required

```
{  
  "name": "Mark",  
  "netid": "fischerM"  
}
```

```
{  
  'name': 'Mark',  
  'netid': 'fischerM'  
}
```

9

JSON

Arrays

- Arrays are defined by square brackets
- Comma separated list of values
- Values can be strings, numbers, boolean, null, objects, or arrays

```
[ 1, 2, 3 ]
```

```
[  
  "one",  
  "two",  
  "three"  
]
```

```
[  
  { "numeral": 1, "name": "one", "odd": true },  
  { "numeral": 2, "name": "two", "odd": false },  
  { "numeral": 3, "name": "three", "odd": true }  
]
```

10

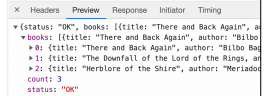
JSON

For Humans

- Many code editors will auto-format JSON for you
- Postman will pretty-print JSON output or show raw
- Some browsers display pretty-print JSON, others render it as an expandable tree



```
{  
  "status": "OK",  
  "books": [  
    {  
      "title": "There and Back Again",  
      "author": "Bilbo Baggins"  
    },  
    {  
      "title": "The Downfall of the Lord of the  
      King",  
      "author": "Frodo Baggins"  
    },  
    {  
      "title": "Merblore of the Shire",  
      "author": "Meriadoc Brandybuck"  
    }  
  ],  
  "count": 3  
}
```



```
{ status: "OK", books: [{ title: "There and Back Again", author: "Bilbo Baggins"}, { title: "The Downfall of the Lord of the King", author: "Frodo Baggins"}, { title: "Merblore of the Shire", author: "Meriadoc Brandybuck"}], count: 3 }
```

11


JSON

In JavaScript

- Language level JSON object
- Can't create instances with `new`, `static` methods only
- JavaScript → JSON

```
let obj = {  
  'books': [  
    {  
      'title': "There and Back Again",  
      'author': "Bilbo Baggins"  
    },  
    {  
      'title': "The Downfall of  
      King",  
      'author': "Frodo Baggins"  
    },  
    {  
      'title': "Merblore of the Shire",  
      'author': "Meriadoc Brandybuck"  
    }  
  ]  
}  
  
console.log( JSON.stringify(obj) )
```

```
JSON.stringify( var )
```



```
{  
  "books": [{"title": "There and Back Again", "author": "Bilbo Baggins"}, {"title": "The Downfall of the Lord of the King", "author": "Frodo Baggins"}]  
}
```

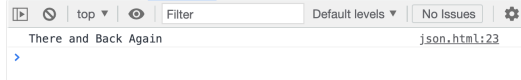
12

JSON In JavaScript

- JSON → JavaScript

```
JSON.parse( string )
```

```
let jsonString = '{"title":"There and Back Again","author":"Bilbo Baggins"}'  
book = JSON.parse( jsonString )  
console.log( book.title )
```



13

JSON In JavaScript

- JSON is always valid JavaScript
- JavaScript is *NOT* always valid JSON
- Example: Trailing commas are fine in JavaScript, but are invalid in JSON

```
let obj = {  
  'books': [  
    {  
      'title': "There and Back A  
      'author': "Bilbo Baggins"  
    },  
    {  
      'title': "The Downfall of  
      'author': "Frodo Baggins"  
    }  
  ]  
}  
console.log( JSON.stringify(obj) )
```

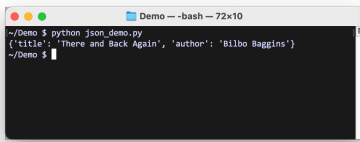
14

JSON In Python

- json module is part of the Python standard library
- JSON → Python

```
import json  
jsonString = '{"title":"There and Back Again","auth  
print(json.loads(jsonString))
```

```
json.loads( var )
```



15

JSON In Python

- Python → JSON

```
json.dumps( var )
```

```
import json

obj = {
    "books": [
        { "title": "There and Back Again", "author": "Bilbo Baggins" },
        { "title": "The Downfall of the Lord of the Rings, and the Return of the King", "author": "Frodo Baggins" },
    ]
}

print(json.dumps(obj))
```

```
Demo --bash-- 72x10
~/Demo $ python json_demo.py
{"books": [{"title": "There and Back Again", "author": "Bilbo Baggins"}, {"title": "The Downfall of the Lord of the Rings, and the Return of the King", "author": "Frodo Baggins"}]}
~/Demo $
```

16

JSON In Python

- Python → JSON
- Optional `indent` argument to `dumps` will pretty-print your JSON strings from Python

```
import json

...

print(json.dumps(obj, indent=2))
```

```
Demo --bash-- 85x16
~/Demo $ python json_demo.py
{
  "books": [
    {
      "title": "There and Back Again",
      "author": "Bilbo Baggins"
    },
    {
      "title": "The Downfall of the Lord of the Rings, and the Return of the King",
      "author": "Frodo Baggins"
    }
  ]
}
~/Demo $
```

17

REST

Representational State Transfer

18

REST

Representational State Transfer

- JSON objects = DB records
- Send & Receive over HTTP
- URLs = object IDs

19

REST

Fundamentals

- REST is not a protocol, like HTTP, or SOAP
- REST is an architectural style, defined by a few key principles

https://en.wikipedia.org/wiki/Representational_state_transfer

20

REST

Client-Server Architecture

- Separation of concerns
- Decouples user interface from data access and persistence
- Allows for many different architectures for client and server

21

REST

Uniform Interface

- Requests should identify resources
 - They do so by using a uniform resource identifier (URI)
- Resource manipulation through representations
 - When a client holds a representation of a resource, including any metadata attached, it has enough information to modify or delete the resource's state
- Self-descriptive messages contain metadata about how the client can best use them
- A REST client should then be able to use server-provided links dynamically to discover all the available resources it needs

22

REST

Statelessness

- Clients can request resources in any order, and every request is stateless or isolated from other requests
- Statelessness refers to a communication method in which the server completes every client request independently of all previous requests
- Implies that the server can completely understand and fulfill the request every time

23

REST

Layered System

- A client can connect to other authorized intermediaries between the client and server, and it will still receive responses from the server
- Design your RESTful web service to run on several servers with multiple layers such as security, application, and business logic, working together to fulfill client requests
- These layers remain invisible to the client

24

REST

Cacheability

- As on the World Wide Web, clients and intermediaries can cache responses
- Well-managed caching partially or completely eliminates some client-server interactions, further improving scalability and performance
- The cache can be performed at the client machine in memory or browser cache storage
- Additionally cache can be stored in a Content Delivery Network (CDN)

25

REST

Semantic HTTP Methods

Method	Description
GET	Get a representation of the target resource's state
POST	Let the host process a resource state sent in the request
PUT	Create or replace the state of a target resource with the state defined in the request
PATCH	Partially update a resource's state
DELETE	Delete the target resource's state
OPTIONS	Describe the available methods

26

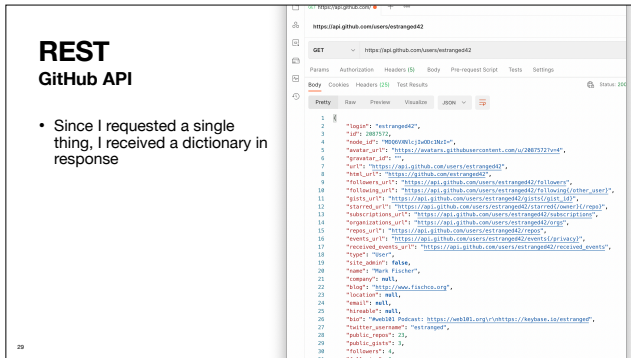
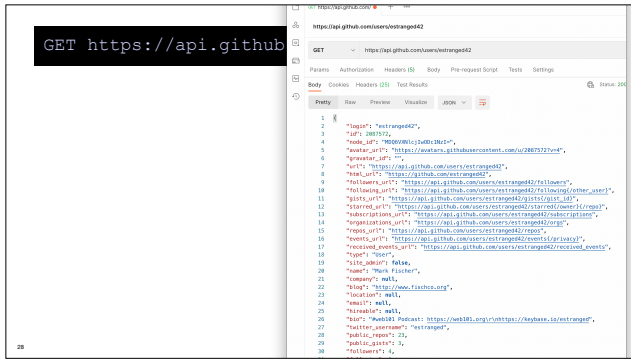
REST

GitHub API

- For example here is the GitHub API call to list basic info about my personal GitHub account

```
GET https://api.github.com/users/estranged42
```

27



REST GitHub API

- Typically all the records in a list will have the same fields, although JSON does not enforce this.

```
{
  "id": 12631784,
  "node_id": "DEwL1jG9zaXVvbnkxMjY5Mjc0NDQg",
  "name": "Adafruit-GFX-Library",
  "full_name": "estranged42/Adafruit-GFX-Library",
  "private": false,
  "html_url": "https://github.com/estranged42/Adafruit-GFX-Library",
  "description": "Adafruit GFX graphics core library, this is the core",
  "fork": true,
  "url": "https://api.github.com/repos/estranged42/Adafruit-GFX-Library",
  "forks_url": "https://api.github.com/repos/estranged42/Adafruit-GFX-Library/forks",
  "parent": {
    "id": 12182878,
    "node_id": "DEwL1jG9zaXVvbnkxMjY5Mjc0NDQgMjE",
    "name": "Adafruit_TinyMPU6121",
    "full_name": "estranged42/Adafruit_TinyMPU6121",
    "private": false,
    "html_url": "https://github.com/estranged42/Adafruit_TinyMPU6121",
    "description": null,
    "fork": false,
    "url": "https://api.github.com/repos/estranged42/Adafruit_TinyMPU6121",
    "forks_url": "https://api.github.com/repos/estranged42/Adafruit_TinyMPU6121/forks",
    "parent": null
  },
  "language": "C++"
},
{
  "id": 12444932,
  "node_id": "DEwL1jG9zaXVvbnkxMjY5Mjc0NDQgMjE",
  "name": "arduino-status-screen",
  "full_name": "estranged42/arduino-status-screen",
  "private": false,
  "html_url": "https://github.com/estranged42/arduino-status-screen",
  "description": "LCD Status Screen with an Adafruit HU22A332",
  "fork": false,
  "url": "https://api.github.com/repos/estranged42/arduino-status-screen",
  "forks_url": "https://api.github.com/repos/estranged42/arduino-status-screen/forks",
  "parent": null
}
```

31

REST GitHub API

- Typically records will have some sort of unique identifier

```
{
  "id": 12631784,
  "node_id": "DEwL1jG9zaXVvbnkxMjY5Mjc0NDQg",
  "name": "Adafruit-GFX-Library",
  "full_name": "estranged42/Adafruit-GFX-Library",
  "private": false,
  "html_url": "https://github.com/estranged42/Adafruit-GFX-Library",
  "description": "Adafruit GFX graphics core library, this is the core",
  "fork": true,
  "url": "https://api.github.com/repos/estranged42/Adafruit-GFX-Library",
  "forks_url": "https://api.github.com/repos/estranged42/Adafruit-GFX-Library/forks",
  "parent": {
    "id": 12182878,
    "node_id": "DEwL1jG9zaXVvbnkxMjY5Mjc0NDQgMjE",
    "name": "Adafruit_TinyMPU6121",
    "full_name": "estranged42/Adafruit_TinyMPU6121",
    "private": false,
    "html_url": "https://github.com/estranged42/Adafruit_TinyMPU6121",
    "description": null,
    "fork": false,
    "url": "https://api.github.com/repos/estranged42/Adafruit_TinyMPU6121",
    "forks_url": "https://api.github.com/repos/estranged42/Adafruit_TinyMPU6121/forks",
    "parent": null
  },
  "language": "C++"
},
{
  "id": 12444932,
  "node_id": "DEwL1jG9zaXVvbnkxMjY5Mjc0NDQgMjE",
  "name": "arduino-status-screen",
  "full_name": "estranged42/arduino-status-screen",
  "private": false,
  "html_url": "https://github.com/estranged42/arduino-status-screen",
  "description": "LCD Status Screen with an Adafruit HU22A332",
  "fork": false,
  "url": "https://api.github.com/repos/estranged42/arduino-status-screen",
  "forks_url": "https://api.github.com/repos/estranged42/arduino-status-screen/forks",
  "parent": null
}
```

32

REST GitHub API

- There are specific URLs for each individual repository

```
{
  "id": 15151009,
  "node_id": "MDUwL1JlG9zaXVvbnkxMjY5Mjc0NDQg",
  "name": "ArduinoCore-samd",
  "full_name": "estranged42/ArduinoCore-samd",
  "private": false,
  "html_url": "https://github.com/estranged42/ArduinoCore-samd",
  "description": "Arduino Core for SAMD21 CPU",
  "fork": true,
  "url": "https://api.github.com/repos/estranged42/ArduinoCore-samd",
  "forks_url": "https://api.github.com/repos/estranged42/ArduinoCore-samd/forks",
  "parent": {
    "id": 3742349,
    "node_id": "MDUwL1JlG9zaXVvbnkxMjY5Mjc0NDQg",
    "name": "ArduinoCore-samd",
    "full_name": "arduino/ArduinoCore-samd",
    "private": false,
    "html_url": "https://github.com/arduino/ArduinoCore-samd",
    "description": null,
    "fork": false,
    "url": "https://api.github.com/repos/arduino/ArduinoCore-samd",
    "forks_url": "https://api.github.com/repos/arduino/ArduinoCore-samd/forks",
    "parent": null
  },
  "language": "C++"
}
```

GET https://api.github.com/repos/estranged42/ArduinoCore-samd

```
{
  "status": 200,
  "headers": {}
}
```

33