JSON Schema Validation

JSON Schema Validation

- Defining the structure of JSON data
- JSON Schema is a vocabulary that allows you to annotate and validate JSON documents
- Most basic schema is a blank JSON object {}
- { "type": "object" }

What is JSON?

JSON

Advantages

- Easy to parse

```
{
  "name"     : "Ray Villalobos",
  "position" : "Staff Author",
  "courses"  : [
    "JavaScript and AJAX",
    "Building Facebook Applications",
    "jQuery Mobile Web Apps"
   ]
}

var info = JSON.parse(data);
info.courses[1]
```

```
{
  "checked": true,
  "dimensions": {
    "width": 5,
    "height": 10
  },
  "id": 1,
  "name": "A green door",
  "price": 12.5,
  "tags": [
    "home",
    "green"
  ]
}
```

What we can check in JSON Schema

1. We can check for the **type of fields** and fail the test if Type is different
2. We can check for **required fields**
3. We can check the number of **min propteroes should come in response**
4. We can verify with example data.

What we can check in JSON Schema

5. If we have additionalProperties : false, So Schema validation will fail if we have extra key added to response.

6. banUnknownProperties is will the test if you have unknown or extra field like additonalProperties

"7. You can use the Regex also in Properties like to verify field data type (e.g I am checking if property proBinaryName has .exe or not)

```
{
  ""type"": ""object"",
  ""properties"": {
    ""progBinaryName"": {
      ""type"": ""string"",
      ""pattern"": ""^[A-Za-z0-9 -_]+_Prog\\.(exe|EXE)$""
    }
  }
}"
```

How to do it in Postman

1. Copy the response to these sites

// Create the Schema using jsonschema.net

// another tool - https://easy-json-schema.github.io/

// https://www.liquid-technologies.com/online-json-to-schema-converter"

2. create a variable like this

var schema = {}; put into it

3. Use Tiny validate to verify with reponse data.

```
pm.test('JSON Schema', function() {
  var jsonData = pm.response.json();
  pm.expect(tv4.validate(jsonData, schema)).to.be.true;
});
```

How to do it in Postman

In POSTMAN

https://www.getpostman.com/collections/e41482d4eb033b7c47b7

JSON Schema

- {}
- { "type": "string" }
- { "$schema": "http://json-schema.org/schema#" }
- "minLength": 2,
- "maxLength": 3
- "pattern": "^(\\([0-9]{3}\\))?[0-9]{3}-[0-9]{4}$"
- Properties
- Required
- items

JSON Schema

https://www.jsonschemavalidator.net/

https://www.jsonschemavalidator.net/s/hG6wNqk4

https://ajv.js.org/keywords.html

https://www.jsonschema.net/app/schemas/0

https://www.jsonschemavalidator.net/

How to do in REST ASSURED?

```
<dependency>
   <groupId>io.rest-assured</groupId>
   <artifactId>json-schema-validator</artifactId>
   <version>4.3.1</version>
</dependency>
```

Adding maven dependency of json-schema-validator

How to do in REST ASSURED?

```
<dependency>
   <groupId>io.rest-assured</groupId>
   <artifactId>json-schema-validator</artifactId>
   <version>4.3.1</version>
</dependency>
```

Adding maven dependency of json-schema-validator

```
import io.restassured.module.jsv.JsonSchemaValidator;

JsonSchemaValidator.matchesJsonSchemaInClasspath("AuthJsonSchema.json")
```

How to do in REST ASSURED?

https://github.com/PramodDutta/JsonSchemaValidationRestAssuredDemo

https://reqres.in/api/users/2