Day 16 - JSON

JSON (JavaScript Object Notation)

JSON syntax is derived from JavaScript object notation syntax, just JSON format is text or string only. JSON is a lightweight data format for storing and transporting. Mostly used when data is sent from server to client. Easier-to-use alternative to XML.

Example:

```
{
    "users":[
        {
            "firstName":"Chris",
            "lastName":"Guerrero",
            "age":250,
            "email":"chris@chris.com"
        },
        {
            "firstName":"Albus",
            "lastName":"Dumbledore",
            "age":70,
            "email":"dumbledore@hogwarts.com"
        },
        {
            "firstName": "Harry",
            "lastName": "Potter",
            "age": 19,
            "email": "potter@hogwarts.com"
        }
    ]
}
```

Above JSON example isn't very different from a normal object. The difference is the key of a JSON object should be with double quotes or should be a string. A JS object and JSON are very similar in that we can change JSON to Object and Object to JSON.

```
{
        "Alex": {
            "email": "alex@alex.com",
            "skills": [
                "HTML",
                "CSS",
                "JavaScript"
            ],
            "age":19,
            "isLoggedIn": false,
            "points": 30
        },
        "Chris": {
            "email": "chris@chris.com",
            "skills": [
                "HTML",
                "JavaScript"
            ],
            "age": 24,
            "isLoggedIn": true,
            "points": 31
        }
    }
```

Converting JSON to JavaScript Object

We mostly fetch JSON data from HTTP response or from a file, but JSON can also be stored as a string and we can change it to an object for demonstration. In JS, the keyword JSON has *parse()* and *stringify()* methods. If we want to change JSON to an object, we parse the JSON using *JSON.parse()*. If we want to change the object to JSON we use *JSON.stringify()*.\

JSON.parse()

```
JSON.parse(json[, reviver])
// json or text, the data
// reviver is an optional callback function
/* JSON.parse(json, (key, value) => {

})
*/
```

In the example below, we parse the JSON string to convert it to an Object.

```
const usersText = `{
    "users":[
      {
        "firstName":"Asabeneh",
        "lastName":"Yetayeh",
        "age":250,
        "email":"asab@asb.com"
      },
      {
        "firstName":"Alex",
        "lastName":"James",
        "age":25,
        "email":"alex@alex.com"
      },
      {
      "firstName":"Lidiya",
      "lastName":"Tekle",
      "age":28,
      "email":"lidiya@lidiya.com"
      }
    ]
}`

const userObj = JSON.parse(usersText, undefined, 4)
console.log(userObj)
```

Using a reviver function with JSON.parse()

To use the reviver function as a formatter, we put the keys we want to format. Let's say we're interested in formatting the firstName and lastName of the JSON data.

```
const usersText = `{
"users":[
  {
    "firstName":"Asabeneh",
    "lastName":"Yetayeh",
    "age":250,
    "email":"asab@asb.com"
  },
  {
    "firstName":"Alex",
    "lastName":"James",
    "age":25,
    "email":"alex@alex.com"
  },
  {
  "firstName":"Lidiya",
  "lastName":"Tekle",
  "age":28,
  "email":"lidiya@lidiya.com"
  }
]
}`

const usersObj = JSON.parse(usersText, (key, value) => {
  let newValue =
    typeof value == 'string' && key != 'email' ? value.toUpperCase() : value
  return newValue
})
console.log(usersObj)
```

JSON.parse() is very useful. You don't have to pass optional parameters either. You can just use it with the required parameter and still achieve a lot.

Converting Object to JSON

We we want to change an object to JSON, we use JSON.stringify(). The method takes one required parameter and two optional parameters. The replacer is used as filter and the space is indentations. If we don't want to filter out any of the keys from the object, we can just pass undefined.

```
JSON.stringify(obj, replacer, space)
// json or text, the data
// reviver is an optional callback function
```

Converting an object to a string. We'll keep all the keys and use a 4 space indentation:

```
const users = {
    Alex: {
      email: 'alex@alex.com',
      skills: ['HTML', 'CSS', 'JavaScript'],
      age: 20,
      isLoggedIn: false,
      points: 30
    },
    Asab: {
      email: 'asab@asab.com',
      skills: [
        'HTML',
        'CSS',
        'JavaScript',
        'Redux',
        'MongoDB',
        'Express',
        'React',
        'Node'
      ],
      age: 25,
      isLoggedIn: false,
      points: 50
    },
    Brook: {
      email: 'daniel@daniel.com',
      skills: ['HTML', 'CSS', 'JavaScript', 'React', 'Redux'],
      age: 30,
      isLoggedIn: true,
      points: 50
    },
    Daniel: {
      email: 'daniel@alex.com',
      skills: ['HTML', 'CSS', 'JavaScript', 'Python'],
      age: 20,
      isLoggedIn: false,
      points: 40
    },
    John: {
      email: 'john@john.com',
      skills: ['HTML', 'CSS', 'JavaScript', 'React', 'Redux', 'Node.js'],
      age: 20,
      isLoggedIn: true,
      points: 50
    },
    Thomas: {
      email: 'thomas@thomas.com',
      skills: ['HTML', 'CSS', 'JavaScript', 'React'],
      age: 20,
      isLoggedIn: false,
      points: 40
    },
    Paul: {
      email: 'paul@paul.com',
      skills: [
        'HTML',
        'CSS',
        'JavaScript',
        'MongoDB',
        'Express',
        'React',
        'Node'
      ],
      age: 20,
      isLoggedIn: false,
      points: 40
    }
  }

  const userString = JSON.stringify(users, undefined, 4);
  console.log(userString)
```

Using a Filter Array with JSON.stringify()

Let’s use the replacer as a filter. The user object has a long list of keys but we are interested only in a few of them. We put the keys that we want to keep in an array as shown below and use it in the place of the replacer.

```
const user = {
    firstName: 'Chris',
    lastName: 'Guerrero',
    country: 'USA',
    city: 'Phoenix',
    email: 'chris@chris.com',
    skills: ['HTML', 'CSS', 'JavaScript', 'React', 'Python'],
    age: 300,
    isLoggedIn: true,
    points:50
}

const txt = JSON.stringify(user, ['firstName', 'lastName', 'country', 'city', 'skills'])
console.log(txt)
```