JSON ⇒ Javascript object notation

→ JSON use double quoty only
⇒ You can write JSON with Javascript object
⇒ Javascript can include function which not available in JSON

What is json

→ start & ends with {}, name value pair seperated by colon (:) ~~, more than one pair of~~ can seprated by (,) comma

file.json

```
{
  "name": "ghansham",
  "mis": 111903033,
  "edu": [ { "year": 2017, "Post" = "10th" },
           { "year": 2019, "post" = "12th" },
           { "year": 2023 , "post" = "graduation" }
         ]
}
```

(name = key, value)

array of objects

→ It is common data format for browser largely replacing XML used by AJAX

Resource

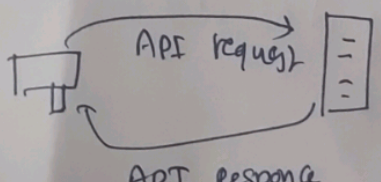
⇒ JSONlint.com ⇒ used to validate json code

⇒ myjson ⇒ A storage for json

⇒ randomuser ⇒ A api which gives random json data

for practise purpose we use it

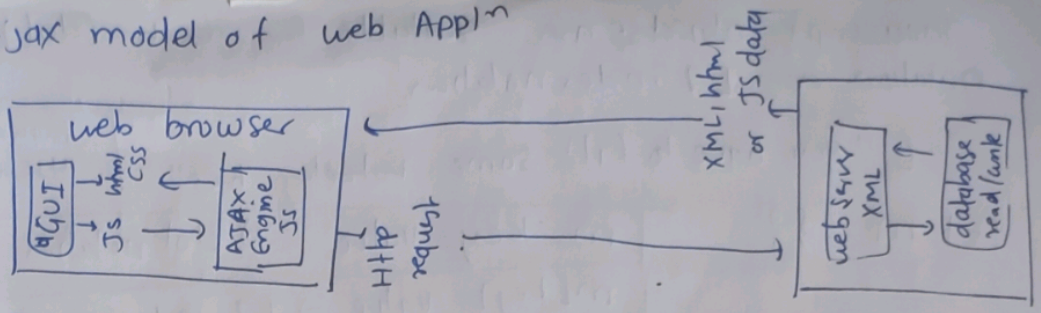
Data structure ⇒ we have used many ds to store content



→ APIs are made up of requests and responcy

CAJAX

Ajax model of web Appln



Data types in JSON

⊙ Number ⊙ String: strictly double quotes (" ") ⊙ Boolean
Array, object (key value pair), Null (just empty)

"name" : "null" → we don't have any function in JSON

{ "car1" : "black",
"car2" : yellow", } ⇒ these are two diffrat objects
}

⇒ JSON VS XML VS YAML ⇒ do a search

{"hell" : "yes"} → JSON object → only double quotes

Var O = {"hell" : "yes"} Javascript object → also can use double or single quotes

⇒ JSONeditoronline.org ⇒ to code JSON

⇒ address : {
"streetname" : value,
"city" : "value",
"state" : "value"
} ⇒ Such like that we can arrange data
}

⇒ If you want to test your code just copy object paste in JS and see & visualize it.

How to do object declaration in JS

var m = {} m.car1 = "black" m.car2 = "blue" one way	var m = {} m["car1"] = "black" m["car2"] = "blue" another way	var m = { "car1" : "black", "car2" : "blue", } another way

Javascript objects are suitable for both (dot). notation or [] index notation

["you have to fill some ~~value~~key here" | Both are useful

let m = {}

m.key = value ⇒ dot notation

m[key] = value ⇒ [] index notation

JSON arrays

* search about JSON in mozilla developer website

① loop through object

for each

```
People.foreach(function(index){
    console.log(people[index])
})
```

↑ object array

this will print key value pair

```
people => obj
for(let i in people){
    console.log(people[i])
}
```

↑ thus you can run loop through JSON object

Also you can run for a object in object

```
people.foreach(()index)=>{
    people[index].foreach((k)=>{
        console.log(people[index][k])
    })
})
```

⇒ such like that you can print result

```
for(let ia in people){
    for(let j in people[i])
    console.log(people[i][j])
}
```

⇒ thus like that. you can print objects

⇒ in JS you can write function inside object but (Json) not supports ~~JSO~~ function it is strictly data

Adding content to objects

in javascript

say you have a object. and it contain array

```
var obj = {
        People : [ { —||— } → name, surname,
                 { —||— },
        ],
        location : [ { —||— } → state, city
                 { —||— }
                 ]
}
```

you can use all array functions inside Json object in JS

push pop will output as index

So we can use array function to put value

Ex: obj.people.push({"name":"grs","surname":"null"})

↑ and

obj.location.push({"state":"andra","city":"banglore"})

↳ output

such like that we can add data in Javascript object.

Javascript JSON method

these are prebuild javascript method to handle JSON files

(A) JSON.stringify()

convert JS value to JS string

see

```
let s = {"name":"raj","roj":"some"}
let m = JSON.stringify(s)
console.log(m)
```

→ such like

```
"{"name":5,"y":6}"
```

(B) JSON Parse

Parse a JSON string constructing javascript value or object described by the string

Parse → JS value

string ← Stringify

try JSON.parse() & json.stringify()

* the parse & stringify both functions are opposite of each other
⇒ you can't stringfy objects containing function
output of stringfy ⇒ (string) ⇒ you can use string function on it !!

stringify & parse are really helpfull

① functions in (JS) k = JSON.stringfy(object variable)

localStorage.setItem("name you want to give", stringified object)
↑ to store info can ↑ strictly string

var m = localStorage.getItem("name") that's it
↑ string

now if we want use it first convert it into object again

var og = JSON.parse(m)
↑ og is object

⇒ thus we can store info in localstorage & extract it whenever needed it.

② Javascript fetch

Fetch ~~API~~ used to interact with some api and pull back json respance and ~~make it use~~ we use json data for our need

const url = "https://api.myjson.com"
↑ some api website

fetch(url).then(function(respance){

in this you can do whatever you want to do with rspare

}).then(function(data){ you deal with data ;})

try it block of fetch

fetch(url).then(function(responce){ ← Important else you not get objects

return responce.json().some code → return responce.json()

}).then(function(data){

code

fetch → responce → data

});

randomuser.mi lapi → use this

⇒ fetch(url).then(function(responce){

return responce.json();

}).then(function(data){

you can manipulate code here

});

dot data.result ⇒ to get all objects inside

data

data.result → Imp ← return responce.json()

Errors in fetch

shortend function abc(a,b,c){}

↓

so we can write

var abc = a,b,c ⇒ {

code

abc is function name

}

easy way to create fun

⇒ shorthand

⇒ this used to create functions & a,b,c are parameters

· Catch function used to catch & resolve a error

Ex!

```
fetch(url).then(responce => { return responce.json();
.then(data => { code } 3).catch
        ↳ data.resonce (error => {
```

Code to be run if catch error

}

⇒ Such like you can do exception handling in JS

Java script handling multiple returns / responses & how we can manage them

↑ large results multiple JSON

```
const url = '...'

fetch(url).then(function(res){
        return res.json()
}).then(function(data){
        console.log(data.result)
        loop through result
})
```

{ } JSON

{{ ⇒ multiple
{| Json

⇒ If you do ~~JSON~~ data result you see you have multiple result & you have to fetch them ...