JSON - tutorial

* JavaScript Object Notion – Douglas crockford
* Lightweight format – popular – Text based – Stress free – construct/deconstruct
* Language Independent (.json) – Easy to read (user/prog)

"Realtime server to browser comm"

Key : value pairs , Ordered list of values.

name : value pair ⟶ object, Struct, record, dictionary etc..

ordered value : array, list etc...

```
{
  "year" : "2017",      ⟶ No Specific order.
  "name" : "Steve"
}
```

object

* "Employee" JSON object → Name, ID, designation — Property : Keys

Rules :

1) { } ⟶ Start & end with braces
2) Key fields – must be included within quotes (" ") – double
3) (:) ⟶ seperates Key & value.
4) Key : value pairs ⟶ Seperated using Commas.
5) Value ⟶ String, int, Boolean etc..

array → Ordered collection ⟶ within "[" and "]"

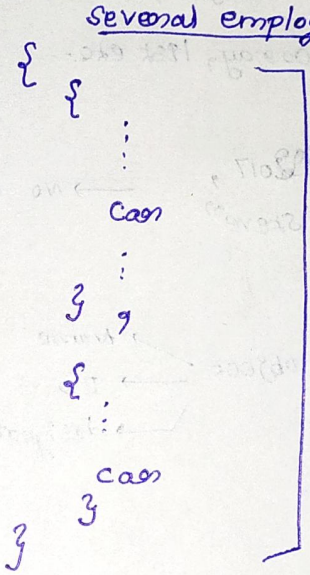
```
{
  "Name" : "Sam",
  "ID" : 5698523,
  "Lang" : ["Java", "python", "C"]
}
```

" " ⟶ empty

```
{
  "Name" : "Sam",
  "ID" : 7234,
  "Role" : "Manager",
  "Lang" : ["Java", "C"],
  "Car" : {
     "Make" : "Maruti",
     "Year" : 2017
  }
}
```

→ Nested JSON

Several employees



3 levels of Nested.

Tool: JSON validation

Why?

* All browsers support - Straightforward syntax
* Object ⟷ text
* API - Usage.
* Fast, less memory, Free, No mapping required., clean,
* JSON - No dependency on other libraries.

Data types
- → Number (real, int, float)
- → String ("")
- → Boolean (true/false)
- → Null (no value)
- → Object → Collection of key-value pair within { }
- → Array: Ordered sequence of values seperated.

* Number: double precision floating-point format
* No hexa/octal

Int, fraction, exponent

String: * → double quotation typing

/ Solidus

\ Reverse Solidus

B Backspace

F Form feed

N Newline

R carriage return

T horizontal tab

U Hexadecimal digits

Boolean: true/false

```
{
  "id": 110,
  ;
}
```
object

```
{ "eBooks" = [
    { "lang": "pascal",
      "ed": 3
    },
    {  :
    },
    {
    },
]
}
```
→ multiple objects.

which book → Index

which lang/ed → using '.'

Asynchronous data calls – no need to refresh.

No name space support
No formal grammar def

JSONLint

JSON editor

JSON minifier

JSON to XML

JSON formatter

---

function
date
undefined
] Also valid datatype in JS.

{ "employees" : [

{ "name" : "Sono" , "mail" : "_____" },

⋮

{ _____ }

] }