JSON

Topics Covered:

- JSON
- JSON Data Types
- JSON Parse
- JSON Stringify
- JSON Objects
- JSON Arrays

Topics in Detail:

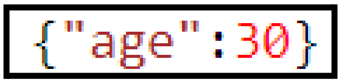
JSON

- **JSON** is the short form of **JavaScript Object Notation**.
- JSON **stores** and **transports data** in **text format**.
- **Using JSON** we can **send data** between computers.
- JSON is **language independent**.

JSON Data Types

- **Numbers**

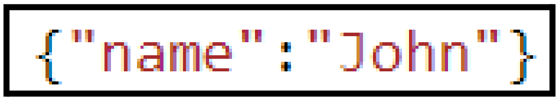
  In JSON, Numbers must be either an **integer** or a **floating-point**.



```
{"age":30}
```

- **String**

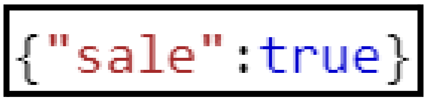
  In JSON, Strings are written **within double quotes**.



```
{"name":"John"}
```

- **Boolean**

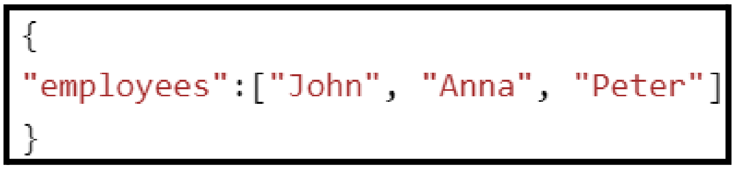
  In JSON, Boolean values can be **either true or false**.



```
{"sale":true}
```

- **Array**

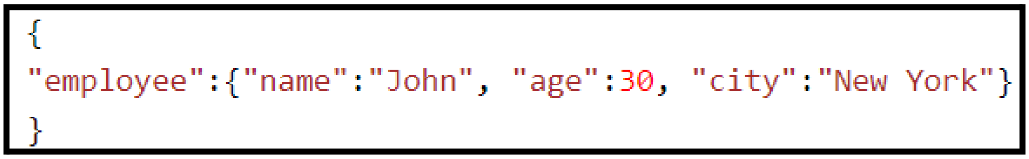
  In JSON, values can be **arrays**.



```
{
"employees":["John", "Anna", "Peter"]
}
```

- **Object**

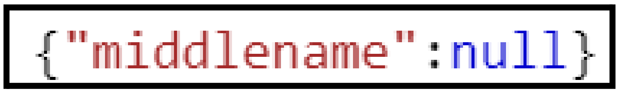
  In JSON, values can be **objects**.



```
{
"employee":{"name":"John", "age":30, "city":"New York"}
}
```

- **null**

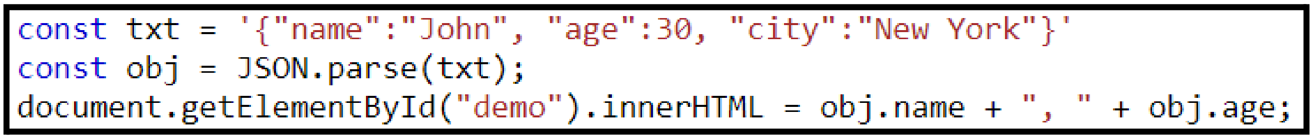
  In JSON, values can be **null**.



```
{"middlename":null}
```

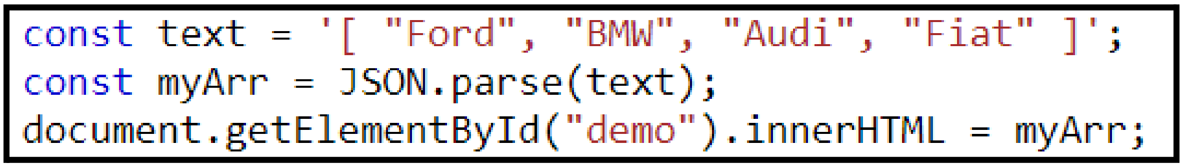
JSON Parse

- The **Data** received from the **web server** is always a **string.**
- **JSON.parse()** method is used to parse the data to a **javascript object**.



```
const txt = '{"name":"John", "age":30, "city":"New York"}'
const obj = JSON.parse(txt);
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age;
```

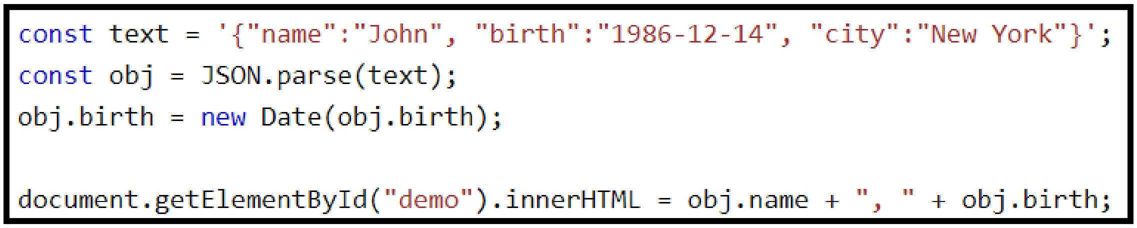
- When **JSON.parse()** is used on an **array**, it will return a **javascript array** instead of a **javascript object**.



```
const text = '[ "Ford", "BMW", "Audi", "Fiat" ]';
const myArr = JSON.parse(text);
document.getElementById("demo").innerHTML = myArr;
```

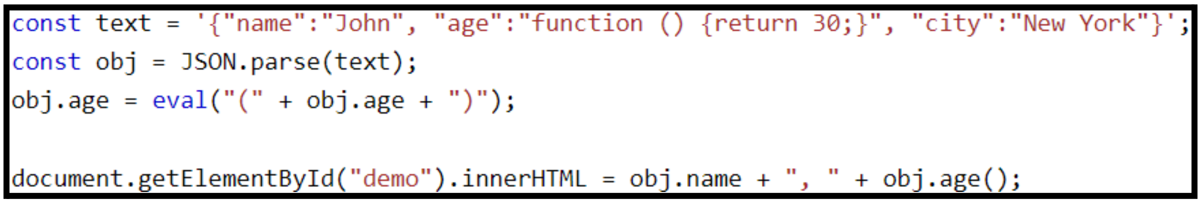
Exceptions

- **Parsing Date**
    - **JSON** does not allow **date format**.
    - So write it as a **string** and **convert** it as a **date object**.



```
const text = '{"name":"John", "birth":"1986-12-14", "city":"New York"}';
const obj = JSON.parse(text);
obj.birth = new Date(obj.birth);

document.getElementById("demo").innerHTML = obj.name + ", " + obj.birth;
```

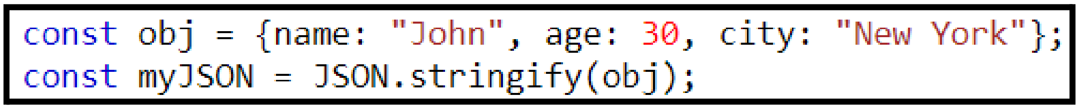
- **Parsing Functions**
    - **JSON** does not allow **Functions**.
    - So write it as a **string** and **convert** it as a **function**.



```
const text = '{"name":"John", "age":"function () {return 30;}", "city":"New York"}';
const obj = JSON.parse(text);
obj.age = eval("(" + obj.age + ")");

document.getElementById("demo").innerHTML = obj.name + ", " + obj.age();
```

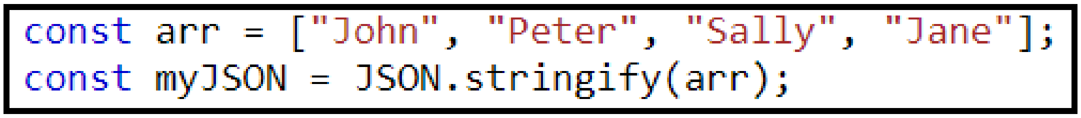
JSON Stringify

- The Data **sent** to the **web server** should be a **string**.
- **JSON.stringify()** method is used to **convert JavaScript objects** into a **string**.



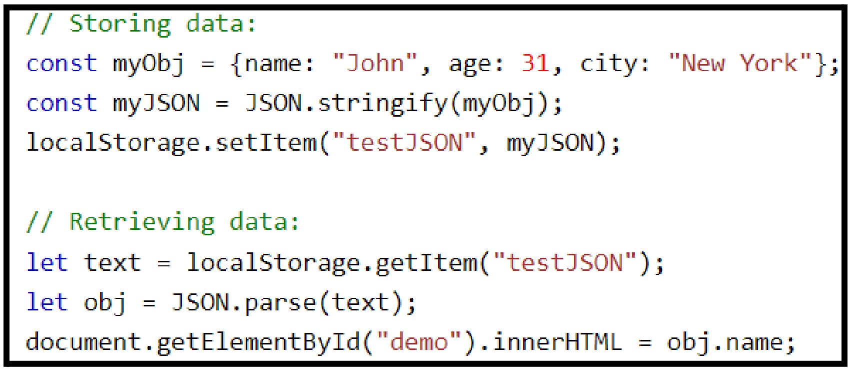
```
const obj = {name: "John", age: 30, city: "New York"};
const myJSON = JSON.stringify(obj);
```

- **JSON.stringify()** methods can also convert **JavaScript Array** into a **string**.



```
const arr = ["John", "Peter", "Sally", "Jane"];
const myJSON = JSON.stringify(arr);
```

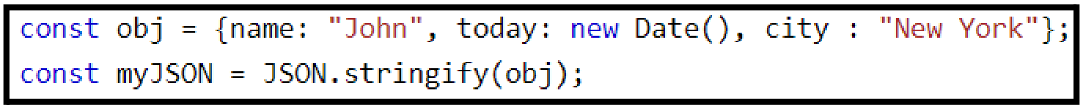
- **Storing Data**
  - In **JSON,** JavaScript objects can be stored as text.



```
// Storing data:
const myObj = {name: "John", age: 31, city: "New York"};
const myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

// Retrieving data:
let text = localStorage.getItem("testJSON");
let obj = JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

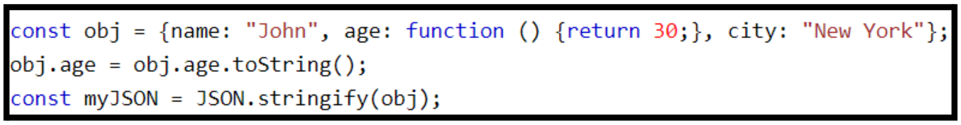
Exceptions

- **Stringify Date**
  - **JSON** does not allow **date objects**.
  - **Date objects** can be **converted** into **strings** using **JSON.stringify()** method.



```
const obj = {name: "John", today: new Date(), city : "New York"};
const myJSON = JSON.stringify(obj);
```

- **Stringify Functions**
  - **JSON** does not allow **Functions** as object values.
  - **Function** from a **javaScript object** will be **removed** while using **JSON.stringify()** method.
  - **Convert functions** into **string** before **JSON.stringify()** method to include functions.

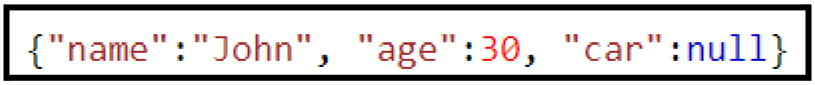


```
const obj = {name: "John", age: function () {return 30;}, city: "New York"};
obj.age = obj.age.toString();
const myJSON = JSON.stringify(obj);
```

JSON Objects

- There is a **JSON object literal** inside every **JSON string**.
- In JSON, **Object literals** are surrounded by **curly braces {}**.
- These object literals contain **key/value pairs**.
- **A colon** separates the keys and values.
- **Keys** must be a **string**.

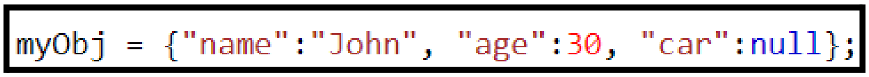
- **Values** should always be **valid JSON Datatype**.
- **A comma** separates each key-value pair.



```
{"name":"John", "age":30, "car":null}
```

Creation of JavaScript Objects

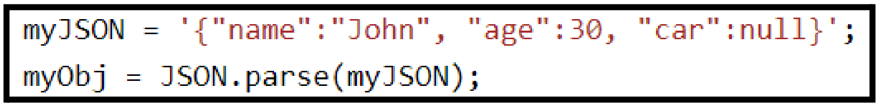
- Javascript objects can be created in two ways
  - From a JSON object literal



```
myObj = {"name":"John", "age":30, "car":null};
```

OR

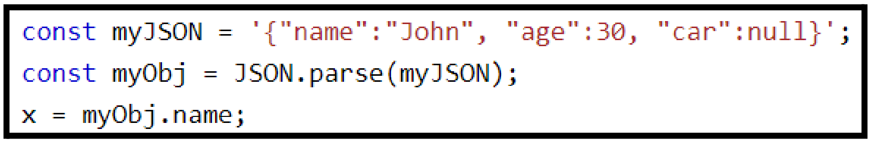
  - By parsing a JSON String



```
myJSON = '{"name":"John", "age":30, "car":null}';
myObj = JSON.parse(myJSON);
```

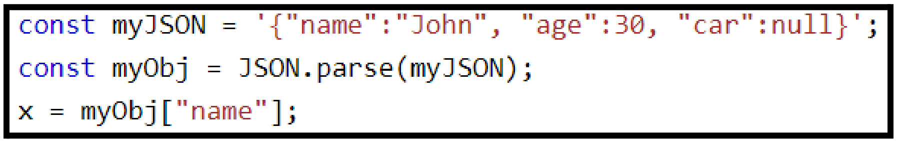
Accessing Object Values

- Object values can be accessed in two ways
  - **The Dot Notation (.)**



```
const myJSON = '{"name":"John", "age":30, "car":null}';
const myObj = JSON.parse(myJSON);
x = myObj.name;
```

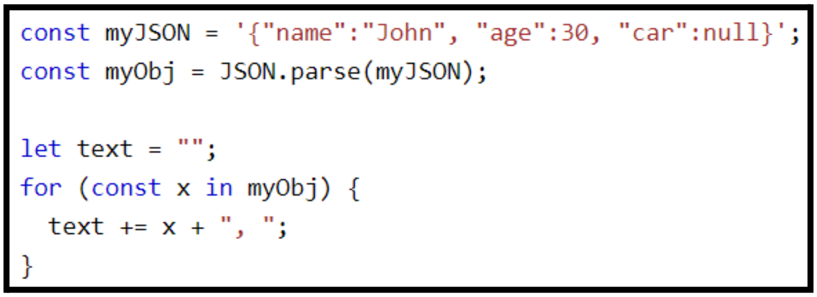
  - **The Array-Like Notation ([ ])**



```
const myJSON = '{"name":"John", "age":30, "car":null}';
const myObj = JSON.parse(myJSON);
x = myObj["name"];
```

Looping an Object

- **for-in loop** is used to loop through object properties.



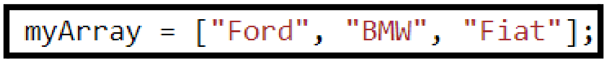
```
const myJSON = '{"name":"John", "age":30, "car":null}';
const myObj = JSON.parse(myJSON);

let text = "";
for (const x in myObj) {
  text += x + ", ";
}
```

JSON Arrays

- Arrays in JSON are similar to JavaScript Array and can have values of the following types
    - string
    - number
    - object
    - array
    - boolean
    - null

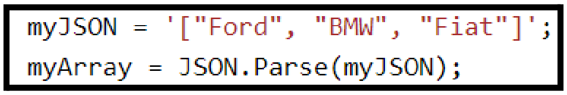
Creating a JSON Array

- Arrays can be created using a **literal**.



```
myArray = ["Ford", "BMW", "Fiat"];
```

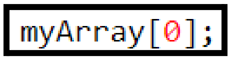
- Arrays can be created by parsing a **JSON string**.



```
myJSON = '["Ford", "BMW", "Fiat"]';
myArray = JSON.Parse(myJSON);
```

Accessing Array Values

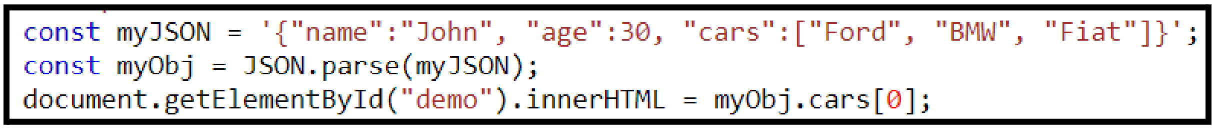
- Array values can be accessed by index.



```
myArray[0];
```

Arrays in Objects

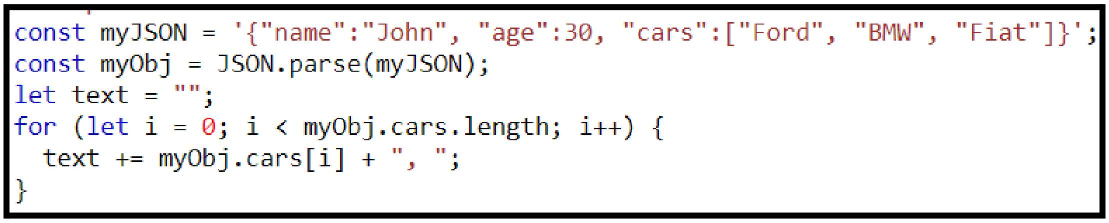
- **Objects** can **contain arrays**, these **array values** can be accessed using an **index**.



```
const myJSON = '{"name":"John", "age":30, "cars":["Ford", "BMW", "Fiat"]}';
const myObj = JSON.parse(myJSON);
document.getElementById("demo").innerHTML = myObj.cars[0];
```

Looping through an array

- The values of the entire array can be accessed by using **for** or **for in** loop.



```
const myJSON = '{"name":"John", "age":30, "cars":["Ford", "BMW", "Fiat"]}';
const myObj = JSON.parse(myJSON);
let text = "";
for (let i = 0; i < myObj.cars.length; i++) {
  text += myObj.cars[i] + ", ";
}
```