05. JSON Objects.

5.1. Object Syntax

Example:

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

- JSON objects are surrounded by curly braces {}.
- JSON objects are written in key/value pairs.
- Keys must be strings, and values must be a valid JSON data type (string, number, object, array, boolean or null).
- Keys and values are separated by a colon.
- Each key/value pair is separated by a comma.

5.2. Accessing Object Values

You can access the object values by using dot (.) notation:

Example:

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

You can also access the object values by using bracket ([]) notation:

Example:

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

5.3. Looping an Object

You can loop through object properties by using the for-in loop:

Example:

```
myObj = { "name":"John", "age":30, "car":null };
for (x in myObj) {
    document.getElementById("demo").innerHTML += x;
}
```

In a for-in loop, use the bracket notation to access the property values:

Example:

```
myObj = { "name":"John", "age":30, "car":null };
for (x in myObj) {
    document.getElementById("demo").innerHTML += myObj[x];
}
```

5.4. Nested JSON Objects

Values in a JSON object can be another JSON object.

Example:

```
myObj = {
    "name":"John",
    "age":30,
    "cars": {
        "car1":"Ford",
        "car2":"BMW",
        "car3":"Fiat"
    }
 }
```

You can access nested JSON objects by using the dot notation or bracket notation:

Example:

```
x = myObj.cars.car2;
//or:
x = myObj.cars["car2"];
```

5.5. Modify Values

You can use the dot notation to modify any value in a JSON object:

Example:

```
myObj.cars.car2 = "Mercedes";
```

You can also use the bracket notation to modify a value in a JSON object:

Example:

```
myObj.cars["car2"] = "Mercedes";
```

5.6. Delete Object Properties

Use the delete keyword to delete properties from a JSON object:

Example:

```
delete myObj.cars.car2;
```