Arrays Objects - JSON

1. Arrays Objects

Los arrays de tipo objeto, son otra forma para poder representar arrays, una alternativa a los arrays convencionales.

Lo que diferencia los objects de los asociativos habituales es la forma en como se puede llamar a los datos dentro de ellos.

Diferencias entre los JS Arrays - Arrays Object

Acceder a los datos:

- **Arrays JS Habituales**

Los Array JS se puede acceder a los datos con la forma de;

- Array[IndiceInt]

- **Objects**

A diferencia de los asociativos los arrays tipo object tienen otra forma de acceso bastante interesante;

- Array[KeyString]
- Array.ObjectData

La forma de hacerlo colocando un (.) luego del array, y posteriormente poner el nombre del índice que queremos acceder, en este caso tenemos que acceder colocando la KeyString pero sin necesidad de colocar los corchetes.

Mauricio Martínez - rigobertomauriciomartinez@gmail.com
http://moudevblog.wordpress.com

Declaración:

- **Arrays JS**

Los arrays convencionales solo se pueden declarar con corchetes.

```
var vegetales = ['Repollo', 'Nabo', 'Rábano', 'Zanahoria'];
console.log(vegetales);
// ["Repollo", "Nabo", "Rábano", "Zanahoria"]
```

- **Array Object**

```
let day1 = {
  squirrel: false,
  events: ["work", "touched tree", "pizza", "running"]
};
```

Cambiar Datos

- **Arrays JS**

```
Tipomúsica = new Array(25);
Tipomúsica[0] = "R&B";
Tipomúsica[1] = "Blues";
Tipomúsica[2] = "Jazz";
```

- **Array Object**

```
let object1 = {value: 10};
let object2 = object1;
let object3 = {value: 10};

console.log(object1 == object2);
// → true
console.log(object1 == object3);
// → false

object1.value = 15;
console.log(object2.value);
// → 15
console.log(object3.value);
// → 10
```

-

Eliminar

- **Asociativos**

En los arrays JS se puede hacer con metodos especificos para eso, y modificar la estructura del array al antojo.

Eliminar 0 elementos desde el índice 2 e insertar "drum"

```
var myFish = ['angel', 'clown', 'mandarin', 'sturgeon'];
var removed = myFish.splice(2, 0, 'drum');

// myFish is ["angel", "clown", "drum", "mandarin", "sturgeon"]
// removed is [], no elements removed
```

Eliminar 1 elemento desde el índice 3

```
var myFish = ['angel', 'clown', 'drum', 'mandarin', 'sturgeon'];
var removed = myFish.splice(3, 1);

// removed is ["mandarin"]
// myFish is ["angel", "clown", "drum", "sturgeon"]
```

- **Arrays Object**

Basta con colocar la palabra “delete” más el nombre del elemento que queremos borrar

- ***delete*** *array.KeyString*

Otras observaciones

Las principales entre los arrays convencionales de JS y los tipo object es la capacidad para manipularlos, los convencionales tienen métodos para poderlos modificar directamente, se hace más fácil esta tarea, en comparación de los tipo object.

2. JSON Introducción

¿Qué es JSON?

- JavaScript Object Notation.
- JSON es una sintaxis para almacenar e intercambiar datos.
- JSON es texto, escrito con notación de objetos JavaScript.
- Su extensión de archivo es (.json)

¿Porqué usar JSON?

Como el formato JSON solo es de texto, puede enviarse fácilmente desde y hacia un servidor y utilizarse como un formato de datos mediante cualquier lenguaje de programación.

- **Intercambio de datos**

Al intercambiar datos entre un navegador y un servidor, los datos sólo pueden ser texto. JSON es texto, y podemos convertir cualquier objeto JavaScript en JSON y enviar JSON al servidor. También podemos convertir cualquier JSON recibido del servidor en objetos JavaScript. De esta forma podemos trabajar con los datos como objetos de JavaScript, sin análisis ni traducciones complicadas.

JSON Sintaxis

La sintaxis de JSON es muy similar a los arrays tipo object, donde se necesita declarar un Nombre - Valor. JSON a diferencia de los arrays object de JavaScript, el Nombre, en JSON necesita ir entre comillas dobles (“”).

JSON

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

In JavaScript, you can write string values with double *or* single quotes:

JavaScript

```
{ name:'John' }
```

Tipo de datos en JSON

- String
- Number
- Otro JSON object
- Array
- Boolean
- Null

Los tipos de datos que no puede contener un JSON son aquellos que no están entre comillas dobles, ya que recordando que JSON tiene que ser texto totalmente, y luego con JS manipular los datos.

- Una función
- Fecha (Date)
- undefined

- ***String Type***

Example

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

- ***Number Type***

Example

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

- ***JSON Object***

Example

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

- ***Array***

Example

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

- ***Boolean***

Example

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

- ***Null***

Example

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

Acceder a datos del JSON

Como se mencionó anteriormente los objetos JSON son similares a los arrays tipo object de JS, y para acceder a su valor, tiene que colocarse el nombre del array seguido de un punto el nombre del dato que queremos acceder.

Example

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

Example

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

ou can access nested JSON objects by using

Example

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

Acceder a los datos como un Array

Los JSON son arrays con una estructura específica, y al tratarse de colecciones esto quiere decir que podemos recorrerlos para acceder a los datos almacenados dentro de él.

Example

```
myObj = {
    "name":"John",
    "age":30,
    "cars": [
        { "name":"Ford", "models":[ "Fiesta", "Focus", "Mustang" ] },
        { "name":"BMW", "models":[ "320", "X3", "X5" ] },
        { "name":"Fiat", "models":[ "500", "Panda" ] }
    ]
}
```

```
for (i in myObj.cars) {

    console.log(myObj.cars[i].name);

    for (j in myObj.cars[i].models)
    {
        console.log(myObj.cars[i].models[j] );
    }
}
```

Conversiones de JSON

- JSON - JS Array Object

Cuando recibimos objetos JSON de respuesta por parte del servidor tenemos que manipularlos en JS, para tal motivo tenemos que hacer una conversión.

- **JSON.parse() :** Convierte JSON a Array Object de JS

```
var obj = JSON.parse('{ "name":"John", "age":30, "city":"New York"}');
```

-JS Array Object - JSON

A la hora de enviar datos al servidor, tenemos que hacerlo en algunos casos con formato JSON, por lo cual tenemos que hacer la respectiva conversión de Object a JSON.

- **JSON.stringify()** : Convierte un array object de JS a un objeto JSON para poder enviarlo al servidor y hacer los respectivos procesos.

```
var obj = { "name":"John", "age":30, "city":"New York"};
```

Jse the JavaScript function JSON.stringify( ) to convert it into a string.

```
var myJSON = JSON.stringify(obj);
```

Referencias.

- https://eloquentjavascript.net/04_data.html
- https://www.w3schools.com/js/js_json_intro.asp
- https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Array