V17

Vamos a hablar de una estructura, de un formato en realidad, para la transferencia de información muy utilizado hoy día, muy utilizado.Ees importante que lo comprendan se llama “J, S, O, N”, JSON, lo van a escuchar nombrar así, un objeto JSON, un archivo JSON, datos en formato JSON.
Esto corresponde a las siglas Javascript, “JS”, “Javascript Object”, la “O”, y “Notation”, por “notación”, o sea, notación de objetos Javascript. Básicamente, es un formato muy ligero, es decir muy liviano para almacenar y transportar, eso es importante, para transportar, transferir datos entre aplicaciones, entre computadoras, en cualquier lugar del mundo se usa a menudo, cuando los datos se envían desde un servidor a una página web, es auto-descriptivo y es fácil de entender. Si ustedes miran un archivo JSON pueden comprenderlo, van a ver todas las letras alineadas con comillas y comas y llaves, pero lo entenderán básicamente porque es muy parecido a un objeto, precisamente por eso se llama JSON, “Javascript Object Notation”. De hecho, un archivo de tipo de JSON, puede ser, muy rápidamente en una zona en una sola línea, convertido a un objeto literal, sobre el cual Javascript va a poder trabajar. Luego, cuando queremos devolver esos datos transformados o cuando creamos nosotros una estructura de datos, convertimos al objeto con él, sobre el cual si podemos trabajar dentro del Javascript, a través, también, de una sola línea, de una sola instrucción, en un objeto de tipo JSON, lo vamos a ver en el código, porque como siempre digo, se entiende mejor ahí.

Veamos aquí en el código las reglas de sintaxis, la sintaxis JSON es muy similar a la de los objetos en Javascript, la principal diferencia es que los nombres de las propiedades, también van entre comillas. Veamos esto, vamos a crear una variable en la que almacenaremos datos en formato JSON. Vamos a ponerle un nombre así, porque es descriptivo, y ahora voy a copiar, para hacer más rápidos, más expeditivos, aquí un formato que tengo predefinido, noten que esto parece un objeto Javascript, tenemos llaves, propiedades, dos puntos, valores y comas para separar otra propiedad, dos puntos, el valor de esa propiedad, ¿okey?, cerramos llaves.
Bueno, este es un típico JSON.
Empleados, sería el arreglo, que contiene, en este caso, tres objetos, son tres empleados, pero al ponerle comillas a las propiedades, tanto como a los valores y luego haber indicado aquí con “back tics” que todo esto es un texto, bueno, pues, lo estamos asemejando al formato JSON que nosotros recibiríamos desde una API. Hay, por ejemplo, una base de datos de algún tipo que nos enviará la información en este formato, en el formato JSON, que reiteramos, es muy popular, porque es muy liviano y es muy rápida la transmisión de los datos. Ahora, dentro de nuestra aplicación con Javascript, nosotros no podríamos integrar estos objetos, porque de hecho, no son objetos, aquí son simplemente, bueno pues, caracteres. Vamos a verlo. Hagamos un “console log” de JSON y vamos a ver aquí, a la izquierda, que efectivamente esto es una cadena de texto, bien, ¿cómo nos damos cuenta?, bueno, porque no podríamos, por ejemplo, iterar el objeto, ya que no se trata de un objeto todavía, es un objeto pero, en formato JSON. Aquí dentro de nuestro Javascript no podríamos hacer, por ejemplo, “JSON, punto, empleados”, esto nos daría un error, “undefined”, no existe tal cosa, no existe una propiedad del objeto JSON como tal, esto es simplemente texto. Para reconvertir estos datos, que ya están preparados, por otra parte, para hacerlo, verán que es muy rápido, reitero.

Entonces, para reconvertir estos datos en formato JSON a un objeto Javascript válido, con el que podamos interactuar y trabajar, vamos a utilizar la función o el método “parse” del objeto JSON, podríamos hacerlo así, creemos una constante que se llame, bueno, “data parceada” o “convertida” o, simplemente ,“data”, son nuestros datos y aquí vamos a correr el método “parse” de JSON y se lo vamos a aplicar a esta información, ¿si?, reitero, esto es una, simplemente, una variable.
Aquí el nombre podría ser cualquiero, cualquiera, esto es la data que nosotros recibimos, la información que recibimos en formato JSON, supuestamente, ¿verdad?, lo estamos simulando desde una API externa, desde un recurso externo, bien, o desde una fuente externa. Bien, en este caso para ser más técnicamente prolijos, diríamos que este es el recurso que nos entregan la información desde una fuente externa, okey, vamos a convertir esto en un objeto iterable Javascript, recuerden, esta línea nos muestra cómo está la información cuando llega formato JSON, no podemos trabajar con él y, ¿qué ocurriría ahora si, por ejemplo, nosotros hiciéramos un “console log” de, bueno, “data”, que es nuestra información parceada? Noten la diferencia en la línea de salida la línea seis, nos muestra una cadena de texto, aunque parece un objeto de Javascript, todavía no lo es, es una cadena de texto, un JSON, pero ya en la línea nueve, cuando hemos ejecutado el “JSON parse”, noten que tenemos un “array”, un arreglo con empleados, ¿okey?, tres objetos, por lo tanto, esto es iterable, con esto sí podemos trabajar en nuestra aplicación, por ejemplo, podríamos ingresar un empleado nuevo, vamos a contratar personal bien, hay que dar trabajo, eso es muy bueno, “data”, punto, aquí sería, “punto empleado”, ¿verdad?, empleados porque allí es donde están nuestros, nuestros datos, en “data, punto, empleados” y aquí hagamos un “push” y bueno, contratamos a alguien. Este es el formato no, nombre, perdón, esto tiene que recibir un objeto, estamos hablando de objetos, ahora sí, cuando los hemos convertido, entonces va a recibir una propiedad, aquí ya es un objeto, por lo tanto aquí no hay comillas, en las propiedades, sí en los valores, “Carolina” y su apellido podría ser “Stanley”, okey, vamos a coontratar.
Si nosotros ahora volvemos a preguntar por nuestro objeto “data”, “console log, data”, punto, empleados”. Veremos aquí que accedemos al arreglo empleados dentro del objeto data, que no tiene tres valores como antes, sino cuatro, porque aquí está nuestra última contratación, y podemos borrar, modificar, bueno, en fin, ustedes ya lo saben a eso. Podemos hacer lo que nosotros necesitemos sobre la base de este objeto y una vez que hacemos las transformaciones, tenemos que devolver los datos modificados, para que queden almacenados aquí, en JSON que, reiteramos, esto en la vida real, no sería una variable aquí en nuestro código, sino que estaría entrando desde un recurso externo.
Y ahora enviaríamos los datos modificados, pero no están en formato JSON, y ahora tenemos que convertirlos otra vez a este formato para que nuestra API y nuestra base de datos, pueda interpretarlos correctamente, pueda recibirlos y almacenados, almacenarlos.
Entonces, recapitulemos, cuando ingresan a nuestra aplicación, están en formato JSON, esto es simplemente texto, los convertimos a un objeto Javascript válido, trabajamos con el objeto y para devolverlo, tenemos que reconvertirlo a su formato original.
Esto lo hacemos tan fácil como hicimos esto, simplemente así, vamos a indicar que JSON, es decir, nuestro objeto original, ahora va a ser igual a, bueno, a “data”, “punto – empleados”. Podríamos poner o simplemente “data”, para que viajen todos los datos completos, pero, “data” es un objeto y JSON, aquí, nuestro servicio externo, espera aún un formato JSON, por lo tanto ,vamos a hacer lo siguiente, JSON, punto, y ahora no utilizaremos “parse” porque“parse” convierte de JSON a objeto Javascript. Ahora haremos el camino inverso, de

objeto Javascript a JSON. Eso se hace con la instrucción “stringify”, es bastante descriptiva, ¿no?, convertir a cadena o “stringificar” sería en una traducción incorrecta directamente del inglés, o sea, “JSON stringify data”. Y ahora, si volvemos a mostrar JSON, veremos que es otra vez una cadena, como era al principio, pero tiene el nuevo dato incorporado. Así es como manipulamos los objetos de tipo JSON.