FICHEROS JSON CON JAVA

Java**S**cript **O**bject **N**otation (JSON) es un formato basado en texto estándar para representar datos estructurados en la sintaxis de objetos de JavaScript. Es comúnmente utilizado para transmitir datos en aplicaciones web (por ejemplo: enviar algunos datos desde el servidor al cliente, así estos datos pueden ser mostrados en páginas web, o vice versa). Nos enfrentaremos a menudo con él, así que vamos a ver cómo hacerlo.

Para acceder a un archivo Json podemos acceder como lo hemos hecho hasta ahora, ya que se trata de un fichero de texto, pero vamos a utilizar una librería que nos va a facilitar mucho el trabajo (si ya lo han hecho otros para qué voy a hacerlo yo!!!!!).

Vamos a usar una biblioteca de Java llamada Gson que es de Google, que lo podéis descarga de http://www.java2s.com/Code/Jar/g/Downloadgson222jar.htm

Para poder usar la librería debemos descomprimir el fichero y obtendremos el fichero gson-2.2.2.jar. Ahora tendremos que añadirla en nuestro proyecto. Cuando nuestro proyecto Java necesite bibliotecas JAR (archivo de Java) para funcionar, tenemos que configurar el proyecto para incluir las bibliotecas en la construcción de su ruta. Afortunadamente, Eclipse hace este proceso simple y fácil de recordar.

Creamos una carpeta nueva llamada lib en la carpeta de proyectos. Este nombre es una abreviación de bibliotecas (en inglés: libraries) y contendrá todos los JAR que usaremos en nuestro proyecto. Copiamos el fichero gson-2.2.2.jar en esta carpeta, actualizamos el proyecto para que la carpeta lib sea visible.

Ahora lo que necesitamos es expandir la carpeta lib en Eclipse y para ello debemos seleccionar el fichero jar y con botón derecho seleccionamos “Construir de ruta (Build path)” → “Agregar a construcción de ruta (Add to Build Path)”. El fichero nos aparecerá en “Biblioteca de Referencia (Referenced Libraries)”.

Ahora ya podemos usar la librería cuando la necesitemos.

Para ello la metodología es:

- Debemos crear una cadena (String) con la información a escribir o leer
- Parseamos la cadena a la estructura en la que vamos a guardar el fichero json.

Ejemplo 1: Leer un json con un sólo objeto sin arrays. El fichero datos.json tendrá la siguiente información:

```
{
    "nombre": "Inma",
    "apellidos": "Olias Alvarez",
    "edad": 29
}
```

Hay que leer el fichero y cargarlo en un String. Supongamos que tenemos el fichero cargado en el String llamado fichero.

Para ver cada uno de los campos tendremos que:

```
        // Creamos un objeto Gson

        Gson gson = new Gson();

        // Para obtener cada una de las propiedad del objeto
        // Recordad que no funcionario si tenemos un array.


        Properties properties = gson.fromJson(fichero, Properties.class);

        // Mostrarmos cada una de las propiedades.

        System.out.println(properties.get("nombre"));
        System.out.println(properties.get("apellidos"));
        System.out.println(properties.get("edad"));
```

Aunque lo mejor es que con esos datos que hemos leído creemos una clase, por ejemplo Persona. Para ello tendremos que crear la clase Persona con los atributos nombre (String), apellidos (String) y edad (int). Es decir, los atributos deben ser iguales que los campos del Json, y además del mismo tipo.

Una vez que tengamos la clase para crear un objeto de la clase persona con los datos del fichero:

```
	// Creamos un objeto Gson

	Gson gson = new Gson();

	// Parseamos y creamos la persona

	Persona p1 = gson.fromJson(fichero, Persona.class);

	// fichero sería un String con todos los datos del fichero
	// Presona.class es la clase a la que queremos parsear
```

Bueno, lo normal que es nuestro fichero JSON no tenga sólo un objeto, si no que tenga un array de objetos. Vamos a ver cómo sería para parsear un array de objetos.

Nuestro fichero sería datos_array.json

```
[
  {
    "apellidos": "Olias Alvarez",
    "edad": 29
  },
  {
    "apellidos": "U",
    "edad": 20
  },
  {
    "nombre": "skskskks",
    "apellidos": "sksk skkskskk",
    "edad": 4
  },
  {
    "nombre": "F",
    "apellidos": "U",
    "edad": 20
  }
]
```

Como en el los casos anteriores tenemos que cargar el archivo en un String (lo llamaremos fichero) y una vez que lo tengamos hay que decidir como vamos a guardar el array de objetos, en este caso vamos a hacerlo en un ArrayList.

```
	// Creamos un objeto Gson

	Gson gson = new Gson();

	// Declaramos la variable en donde guardaremos la informacion.
	ArrayList<Persona> personas =gson.fromJson(fichero,
			new TypeToken<ArrayList<Persona>>(){}.getType());

	// TypeToken nos sirve para obtener la clase del objeto a crear.

	// Podemos trabajar ya con el ArrayList
	for (Persona aux : personas) {
		System.out.println(aux.toString());
	}
```

Con esto ya sabemos leer fichero JSON, pero si por ejemplo, modificamos la lista de persona que teníamos a lo largo de la ejecución del programa, nos interesará guardar nuestro ArrayList en un fichero JSON. Vamos a ver cómo escribir fichero JSON a partir de los datos que tengamos en nuestro programa.

```
    // Si tenemos un ArrayList llamado personas, obtenemos el String
    // correspondiente al objeto JSON de personas

    String json = gson.toJson(personas);

    // Y ya tenemos sólo tendremos que escribir ese string
```

Si miramos el fichero veremos nuestro JSON, pero no como estamos acostumbrado a verlo, es decir no aparece “bonito”. Si queremos que aparezca con los saltos de líneas y tabuladores que normalmente utilizamos con los ficheros JSON debemos usar un nuevo constructuro que nos permite obtener un fichero más “bonito”.

```
    final Gson prettyGson = new GsonBuilder().setPrettyPrinting().create();
    final String representacionBonita = prettyGson.toJson(personas);
```

https://www.json.org/json-es.html

https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON

https://www.discoduroderoer.es/leer-y-escribir-json-en-java/