ITEDES

Educación Digital

Módulo:

Fundamentos de Ingeniería de Software

Segmento:

Algoritmos y Estructuras de Datos

Tema:

JSON

Prof. Germán C. Basisty
german.basisty@itedes.com

Índice de Contenidos

Qué es JSON

JSON es el acrónimo para **JavaScript Object Notation**, y aunque su nombre lo diga, no es exclusivamente parte de JavaScript, de hecho es un estándar basado en texto plano para el intercambio de información, por lo que se usa en muchos sistemas que requieren mostrar o enviar información para ser interpretada por otros sistemas. La ventaja de **JSON** al ser un formato que es independiente de cualquier lenguaje de programación, es que los servicios que comparten información por éste método, no necesitan hablar el mismo idioma, es decir, el emisor puede ser por ejemplo **Java**, y el receptor **PHP**. Cada lenguaje tiene su propia biblioteca para codificar y decodificar **JSON**.

Una práctica habitual en los diversos lenguajes de programación, es utilizar aquellas colecciones nativas del propio lenguaje, y codificar a **JSON** cuando haya que compartir esa información con otras aplicaciones. Una excepcion a esta regla es **JavaScript**, que utiliza **JSON** de forma nativa para gestionar sus colecciones de datos.

Tipos de Datos

JSON soporta 3 tipos de datos, a saber:

- Texto (los valores deben encerrarse entre comillas dobles)
- Numérico (números reales)
- Booleanos (true / false)
- También puede ser null, independientemente del tipo de dato.

Formato

JSON puede representar tanto listas como diccionarios, o la combinación de ambos. Los diccionarios se delimitan utilizando llaves ({...}), y las listas utilizando corchetes ([...]). Siempre las claves deben encerrarse entre comillas dobles y deben ser de tipo texto.

Ejemplo básico:

```
{
    "nombre":"Fulano Probencio",
    "edad":27,
    "nacionalidad":"Argentino ",
    "altura":"172 cm",
    "peso":75,
    "pasatiempos":["Futbol ","Handball","Cine","Lectura","Videos"],
    "soltero":true,
    "direccion":{
        "calle":"Ave. Siempre Viva",
        "numero":"123",
        "pais":"México"
    }
}
```

Ejemplo 1

Generar una lista de frutas y un diccionario con los datos de una persona utilizando colecciones nativas. Convertirlos a JSON y mostrar por pantalla.

Python

```python
import json

persona = {}

persona['nombre'] = 'Juan'
persona['apellido'] = 'Perez'
persona['dni'] = 33292147

personaJSON = json.dumps(persona)

print(personaJSON)

frutas = ['uvas', 'manzanas', 'peras', 'naranjas']

frutasJSON = json.dumps(frutas)

print(frutasJSON)
```

JAVA

```
import java.util.HashMap;
import java.util.ArrayList;
import org.json.JSONObject;
import org.json.JSONArray;

public class EjemploJson {
    public static void main(String args[]) {
        HashMap<String, Object> persona = new HashMap<String,
Object>();
        persona.put("nombre", "Juan");
        persona.put("apellido", "Perez");
        persona.put("dni", 33292147);

        JSONObject personaJSON = new JSONObject(persona);

        System.out.println(personaJSON);

        ArrayList<String> frutas = new ArrayList<String>();
        frutas.add("uvas");
        frutas.add("manzanas");
        frutas.add("peras");
        frutas.add("naranjas");

        JSONArray frutasJSON = new JSONArray(frutas);

        System.out.println(frutasJSON);
    }
}
```

Antes de compilar, descargar la biblioteca org.json desde

http://www.java2s.com/Code/JarDownload/java/java-json.jar.zip

y descomprimir el archivo jar (la biblioteca json.org) en el mismo directorio donde esté el código fuente:

```
$ unzip java-json.jar.zip
```

Compilar incluyendo el archivo jar:

```
$ javac -cp java-json.jar:. Ejemplojson.java
```

Ejecutar incluyendo el jar:

```
$ java -cp java-json.jar:. EjemploJson
```

Tener presente que en java se utiliza la clase **JSONObject** para los diccionarios y la clase **JSONArray** para las listas. Se profundizará en esto más adelante.

C#

```
using System;
using Newtonsoft.Json;
using System.Collections.Generic;

namespace json
{
    class Program
    {
        static void Main(string[] args)
        {
            Dictionary<string, string> persona = new
Dictionary<string, string>();
            persona.Add("nombre", "Juan");
            persona.Add("apellido", "Perez");
            persona.Add("dni", "33292147");

            var personaJSON = JsonConvert.SerializeObject(persona);
            Console.WriteLine(personaJSON);

            List<string> frutas= new List<string>();
            frutas.Add("uvas");
            frutas.Add("manzanas");
            frutas.Add("peras");
            frutas.Add("naranjas");

            var frutasJSON = JsonConvert.SerializeObject(frutas);

            Console.WriteLine(frutasJSON);
        }
    }
}
```

Instalar el packete Newsoft.Json utilizando el comando:

```
$ dotnet add package Newtonsoft.Json
```

dentro de la carpeta del proyecto.

JavaScript

```
var persona = {};

persona.nombre = 'Juan';
persona.apellido = 'Perez';
persona.dni = 33292147;

console.log(JSON.stringify(persona));

frutas = [];

frutas.push('uvas');
frutas.push('manzanas');
frutas.push('peras');
frutas.push('naranjas');

console.log(JSON.stringify(frutas));
```

Ejemplo 2

Partiendo de strings en formato JSON, generar una lista de frutas y un diccionario con los datos de una persona utilizando colecciones nativas.

Python

```
import json

jsonString = '{"nombre": "Juan", "apellido": "Perez", "dni":
33292147}'

persona = json.loads(jsonString)

print(persona)

jsonString = '["uvas", "manzanas", "peras", "naranjas"]'

frutas = json.loads(jsonString)

print(frutas)
```

Java

```
import org.json.JSONObject;
import org.json.JSONArray;

public class EjemploJson2 {
    public static void main(String args[]) {
        try {
            JSONObject personaJSON = new
JSONObject("{\"nombre\":\"Juan\",\"apellido\":\"Perez\",\"dni\":332921
47}");
            System.out.println(personaJSON);
        } catch(Exception e) {
            System.out.println("ERROR");
        }

        try {
            JSONArray frutasJSON = new
JSONArray("[\"uvas\",\"manzanas\",\"peras\",\"naranjas\"]");
            System.out.println(frutasJSON);
        } catch(Exception e) {
            System.out.println("ERROR");
        }
    }
}
```

Compilar y ejecutar el ejemplo como el ejemplo 1.

En java no hay una forma amable de convertir un JSONObject en un HashMap / JSONArray a ArrayList, pero es posible trabajar directamente sobre objetos JSON como si fuesen colecciones nativas.

Otra opción es iterar por las claves del JSONObject e ir configurando el HashMap / ArrayList según sea necesario, por ejemplo (ilustrativo):

```
public static HashMap<String, Object> toHashMap(JSONObject object)
throws JSONException {
    HashMap<String, Object> map = new HashMap<String, Object>();

    Iterator<String> keysItr = object.keys();
    while(keysItr.hasNext()) {
        String key = keysItr.next();
        Object value = object.get(key);

        if(value instanceof JSONArray) {
            value = toArrayList((JSONArray) value);
        }

        else if(value instanceof JSONObject) {
            value = toHashMap((JSONObject) value);
        }
        map.put(key, value);
    }
    return map;
}

public static ArrayList<Object> toArrayList(JSONArray array) throws
JSONException {
    ArrayList<Object> list = new ArrayList<Object>();
    for(int i = 0; i < array.length(); i++) {
        Object value = array.get(i);
        if(value instanceof JSONArray) {
            value = toArrayList((JSONArray) value);
        }

        else if(value instanceof JSONObject) {
            value = toHashMap((JSONObject) value);
        }
        list.add(value);
    }
    return list;
}
```

C#

```
using System;
using Newtonsoft.Json;
using System.Collections.Generic;

namespace json
{
    class Program
    {
        static void Main(string[] args)
        {
            string jsonString =
"{\"nombre\":\"Juan\",\"apellido\":\"Perez\",\"dni\":33292147}";

            Dictionary<string, string> persona =
JsonConvert.DeserializeObject<Dictionary<string, string>>(jsonString);

            Console.WriteLine(persona["nombre"]);
            Console.WriteLine(persona["apellido"]);
            Console.WriteLine(persona["dni"]);

            Console.WriteLine();

            jsonString =
"[\"uvas\",\"manzanas\",\"peras\",\"naranjas\"]";

            List<string> frutas =
JsonConvert.DeserializeObject<List<string>>(jsonString);

            foreach(string fruta in frutas)
                Console.WriteLine(fruta);
        }
    }
}
```

JavaScript

```
var persona = '{"nombre":"Juan","apellido":"Perez","dni":33292147}';

console.log(JSON.stringify(persona));

frutas = ["uvas","manzanas","peras","naranjas"];

console.log(JSON.stringify(frutas));
```

Ejercitación

1) Realizar una monografia sobre la biblioteca json.org en Java.

2) Realizar una monografia sobre el paquete Newtonsoft.Json en C#.