Codificar JSON en Java usando org.json

**convertir un objeto de Java a una cadena de JSON**

<https://parzibyte.me/blog/2019/05/09/codificar-json-java-usando-org-json/>

**Parsear un objeto JSON a una clase de Java**

https://parzibyte.me/blog/2019/02/16/parsear-decodificar-json-java-jsonobject-jsonarray**/**

Existen varias librerías para hacer esto JSONObject, GSON, Jackson, etc. Por ejemplo

JSONObject jsonObject = new JSONObject(*objetoJava*);

String codificado = jsonObject.toString();

esto funciona perfecto con objetos java básicos, pero existen algunos inconvenientes cuando los objetos hacen referencias cíclicas, es decir referencian a otros objetos que a su vez referencian a los primeros o directamente son autoreferenciados, existen soluciones.

Para que funcione la clase debe poseer los getters y setters correpondientes a todas sus propiedades (atributos de la clase).

Particularmente tuve inconvenientes con las clasesdefinidas para usar persistencia medianteJPA. En particular aquellas que incluyen referencias cíclicas, por ejemplo, una clase que representa a una tabla que referencia (uno a muchos) a otra tabla, la clase agrega una columna para hacer referencia a otra tabla y esta produce una referencia cíclica y hace que JSONObject de error.

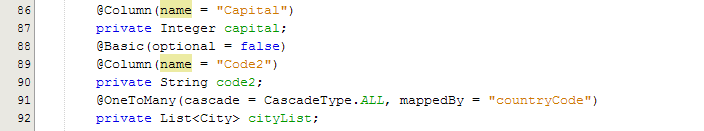
Existe una solución y es indicar mediante una anotación para la columna que produce esa referencia cíclica, de manera de indicar que no sea tenida en cuenta. Cada librería JSON lo trata a su manera.

En particular **JSONObject** lo hace mediante la notación **@JSONPropertyIgnore**. para esto es necesario importar dentro del fuente donde se codifica la clase en cuestión

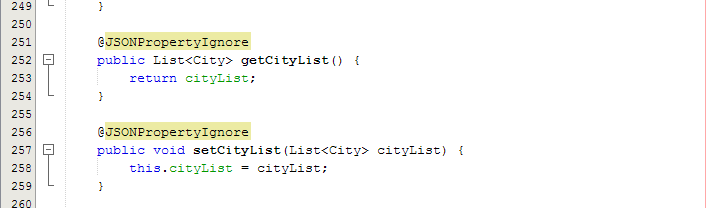
**import org.json.JSONPropertyIgnore;**

aunque la notación no se le pone a la columna, sino que se le agrega a los métodos getter() y setter()

Por ejemplo en la base World tengo Country y City como tablas. Un Country tiene varias Cityies, y en este caso una City pertenece solo a un Country. JPA agrega automáticamente a la tabla Country una columna donde indica la referencia



creó una columna counryCode que es la que utiliza en City para referenciarla. Entonces en el método getter() de esta columna agregar la anotación @JSONPropertyIgnore (por las dudas se la agrege al Setter() pero vi que para esto no era necesario.

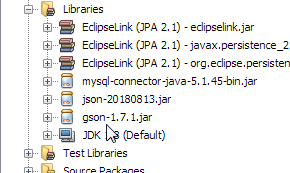


En caso de la librería Jackson utiliza la notación @JsonIgnore en algunas versiones lo antepone a la columan y en otra en el getter()

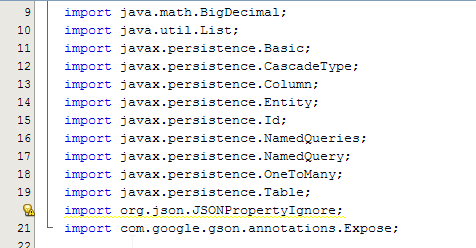
Codificar JSON en Java usando GSON

<https://howtodoinjava.com/gson/gson-exclude-or-ignore-fields/>

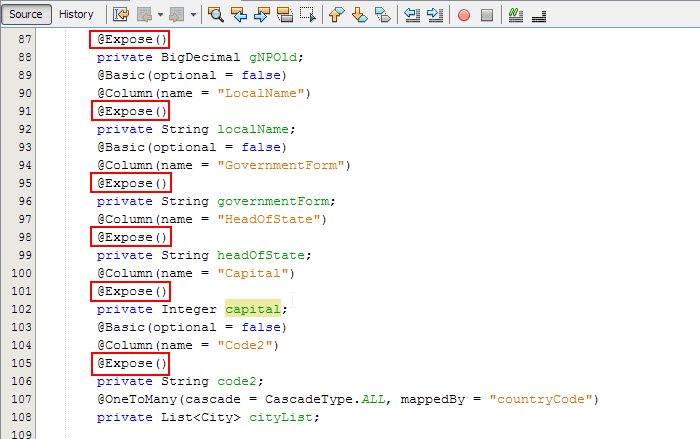
Gson es otra librería para serializar objetos java a Json y al revez. La librería es com.google.gson y si no usamos maven que la incorporaría en el POm.xml, tenemos que agregarla manualmente al la carpeta librería



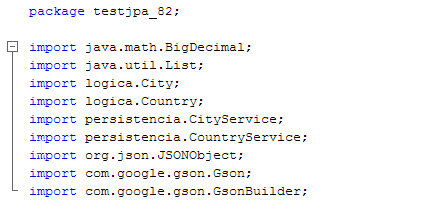
Gson utiliza la anotación @Exclude para indicar que columnas del objeto se seralizan. Las columnas que no contengan dicha anotación serán las que no se serializarán. Para utilizar la anotación debo importarla en el archivo fuente donde se utilizará



En nuestro caso en el objeto Country nos interesa excluir la columna que JPA creo para que lo referencie el objeto City y es la columna que trae el problema ciclico



Por otro lado desde el código donde se va a realizar la serailizacion debemos importar



Generalmente cuando se tragajoa con Gson se crea una instancia del objeto Gson, pero para los casos en que se excluyen columnas se debe crear una instancia del objeto GsonBuilder, como se ve abajo y se llaman a los métodos

Gson gson = new GsonBuilder()

.excludeFieldsWithoutExposeAnnotation()

.create();

