# Manipulação de JSON usando Gson

A melhor fonte acerca de Gson é <https://github.com/google/gson/blob/master/UserGuide.md>. Abaixo segue apenas a ilustração de alguns dos recursos.

## Gerar e obter valores de JSON

A criação de um objeto JSON (e a operação inversa) pode ser realizada conforme ilustrado abaixo.

Gson gson = **new** Gson();

*// toJson*

System.***out***.println(gson.toJson(21));

System.***out***.println(gson.toJson(**true**));

System.***out***.println(gson.toJson(**new** Integer(-57)));

System.***out***.println(gson.toJson(**"hoje"**));

*// fromJson*

System.***out***.println(gson.fromJson(**"\"hoje\""**, String.**class**));

System.***out***.println(gson.fromJson(**"100"**, Integer.**class**));

Nesse exemplos apenas tipos primitivos e sequências de caracteres foram empregados.

## Serialização envolvendo objeto

Abaixo é ilustrado como uma classe arbitrária pode ser serializada e o processo inverso realizado com o emprego de JSON.

**public class** Objeto {

**public static void** main(String[] args) {

Gson gson = **new** Gson();

*// toJson*

String json = gson.toJson(**new** Exemplo(-4, **"Exemplo"**));

*// fromJson*

Exemplo tst = gson.fromJson(json, Exemplo.**class**);

System.***out***.println(json);

System.***out***.println(tst.getValor() + **" "** + tst.getTexto());

}

}

**class** Exemplo {

**private int valor**;

**private** String **texto**;

**public** Exemplo(**int** v, String str) {

**valor** = v;

**texto** = str;

}

**public int** getValor() {

**return valor**;

}

**public** String getTexto() {

**return texto**;

}

}

## Serialização de vetor

Ocorre de forma análoga, conforme ilustrado abaixo.

**public class** Vetor {

**public static void** main(String[] args) {

Gson gson = **new** Gson();

String json = gson.toJson(**new** UmVetor());

System.***out***.println(json);

UmVetor obj = gson.fromJson(json, UmVetor.**class**);

System.***out***.println(obj.**nome**);

System.***out***.println(obj.**logicos**[0]);

}

}

**class** UmVetor {

String **nome** = **"UmVetor"**;

**boolean**[] **logicos** = { **true**, **false**, **false**, **true** };

}

## Serialização de coleções

A serialização é ilustrada abaixo, juntamente com o processo inverso. Observe a necessidade de criação do objeto que descreve o tipo da coleção.

List<Integer> numeros = **new** ArrayList<Integer>();

numeros.add(34);

numeros.add(23);

Gson gson = **new** Gson();

*// toJson*

String json = gson.toJson(numeros);

System.***out***.println(json);

*// fromJson*

Type listaType = **new** TypeToken<List<Integer>>(){}.getType();

List<Integer> recuperado = gson.fromJson(json, listaType);

System.***out***.println(recuperado);