JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON

Filename extension	`.json`
Internet media type	`application/json`
Type code	TEXT
Uniform Type Identifier (UTI)	public.json
Type of format	Data interchange
Extended from	JavaScript
Standard	RFC 7159, ECMA-404
Website	json.org

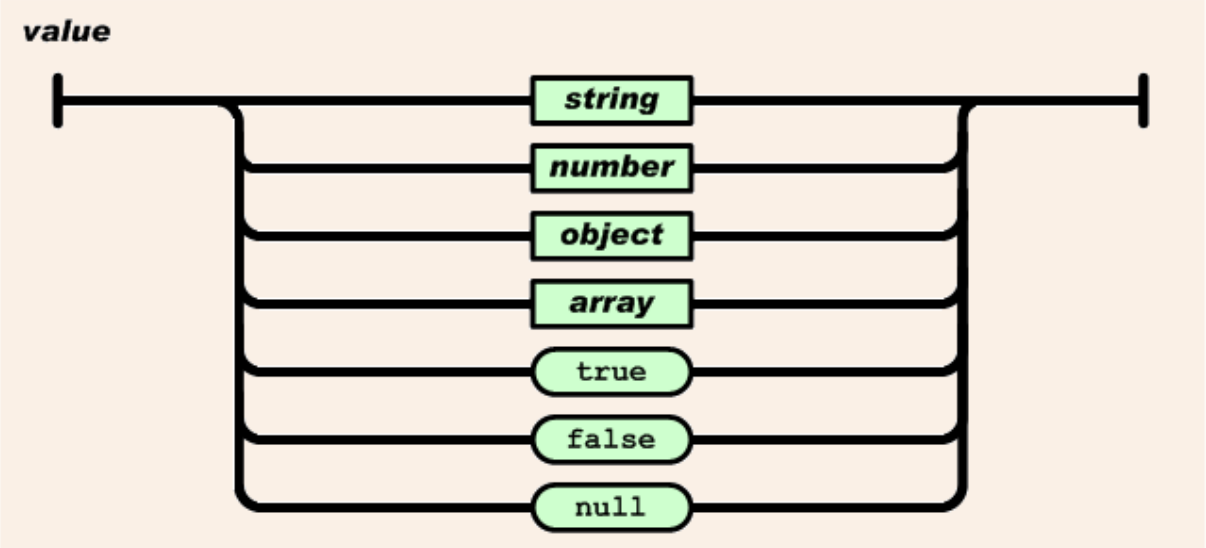
Formát JSON(*Javascript Object Notation*) pôvodne vznikol pre výmenu dát medzi serverovou a klientskou časťou webovej aplikácie. Jedná sa o podmnožinu jazyka Javascript, ktorá dovoľuje reprezentovať základné dátové štruktúry a umožňuje ich priamočiare použitie v prehliadači. Postupom času sa však JSON stal rozšíreným dátovým formátom a knožnice umožňujúce jeho použitie existujú pre všetky nežne používané dátové typy. Použitie v aplikácii je veľmi priamočiare, pretože JSON je možné ľahko mapovať priamo na objekty daného jazyka.

JSON umožňuje reprezentovať 4 jednoduché dátové typy:

reťazec, číslo, logická hodnota a *null*

a 2 štruktúrované:

objekty a polia.



Polia a objekty môžu ako iné svoje prvky obsahovať ľubovoľný ďalší dátový typ – jednoduchý aj štruktúrovaný.

JSON je tak veľmi flexibilný a je v ňom možné reprezentovať v podstate akúkoľvek dátovú štruktúru.

Logické hodnoty: true/false

Špeciálna hodnota: null

true, **false** a **null** se uvádzajú iba malými písmenami

Reťazce: v úvodzovkách, Unicode

Čísla: ako u väčšiny programovacích jazykov pre dátový typ double

Pre oddelenie desatinných miest sa používa bodka, pred číslom je možné uviesť znak(-/+)

Možnosť zapisovať čísla v exponencialnom tvare, napr. milion môžeme zapísať **1e6**

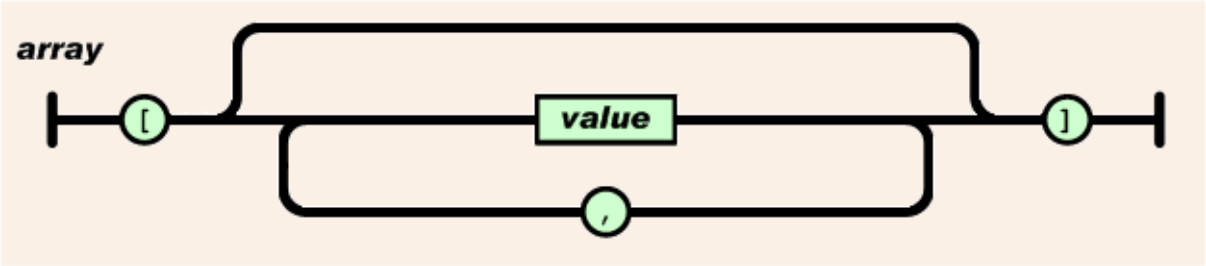
V ostatných jazykoch sa v závislosti na použitom parseri JSON môžu číselné hodnoty mapovať na rôzne dátové typy – celočíselné , desatinné, aj s rôznou presnosťou.

Interoperabilita číselného typu vo formáte JSON

{“n”: 12345678901234567890 }

Javascript -> *7000

Pole



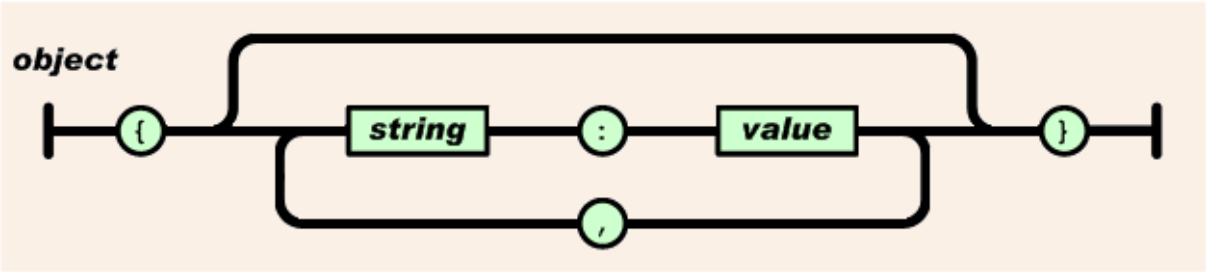
[1, 2, 30]

[true, “ahoj“, 123, “Nazdar“]

Objekt

= neusporiadaná množina dvojíc meno-hodnota

-rôzne dátové typy, aj vnorenie, hodnota – pole



{ “name“: “Ján Novák“,

“age”: 42,

“student”: false

}

Stiahnutie potrebných nástrojov:

http://www.java2s.com/Code/Jar/j/Downloadjsonsimple11jar.htm

http://www.java2s.com/Code/Jar/j/Downloadjacksonall190jar.htm

JSON v Jave

```
public static void main(String[] args) {
    JSONObject obj = new JSONObject();
    obj.put("meno","Milan");
    obj.put("vek", 69);
    JSONArray pole = new JSONArray();
    pole.add("Anna");
    pole.add(1);
    obj.put("zoznam", pole);
    System.out.println(pole.toJSONString());
    System.out.println(obj.toJSONString());
```

JSON -> Java

```
public static void main(String[] args) throws ParseException {
    JSONParser parser = new JSONParser();
   String jsonInString = "{\"name\":\"mkyong\",\"salary\":7500,\"skills\":[\"java\",\"python\"]}";
   Object obj = parser.parse(jsonInString);
  JSONObject objekt = (JSONObject) parser.parse(jsonInString);
    System.out.println(objekt.toJSONString());
}

}
```

Zápis do súboru

```
public static void main(String[] args) {
    FileWriter fw = null;
    try {
        JSONObject object = new JSONObject();
        object.put("meno", "Jan");
        object.put("vek", 45);
        JSONArray deti = new JSONArray();
        deti.add("Janíčko");
        deti.add("Marienka");
        object.put("deti", deti);
        fw = new FileWriter("novy.json");
        fw.write(object.toJSONString());
        fw.flush();
        fw.close();
    } catch (IOException ex) {
        Logger.getLogger(zapisdosuboru.class.getName()).log(Level.SEVERE, null, ex);
    } finally {
        try {
            fw.close();
        } catch (IOException ex) {
            Logger.getLogger(zapisdosuboru.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Zápis zo súboru

```
public static void main(String[] args) {
    try {
        JSONParser parser = new JSONParser();
        Object obj = parser.parse(new FileReader("novy.json"));
        JSONObject jsonObj = (JSONObject) obj;

        String meno = (String) jsonObj.get("meno");
        Long vek = (Long) jsonObj.get("vek");

        JSONArray deti = (JSONArray) jsonObj.get("deti");
        Iterator<String> iterator = deti.iterator();
        while(iterator.hasNext()) {
            System.out.println(iterator.next());
        }

    } catch (FileNotFoundException ex) {
        Logger.getLogger(zoSuboru.class.getName()).log(Level.SEVERE, null, ex);
    } catch (IOException ex) {
        Logger.getLogger(zoSuboru.class.getName()).log(Level.SEVERE, null, ex);
    } catch (ParseException ex) {
        Logger.getLogger(zoSuboru.class.getName()).log(Level.SEVERE, null, ex);
    }

}
```

Úlohy

1. Vytvorte JSON objekt s aspoň 5 atribútmi.
2. Vytvorte JSON objekt, ktorý obsahuje aspoň 2 polia a 2 vnorené objekty.
3. JSON objekt vytvorený v predchádzajúcej úlohe preveďte na Java objekt pomocou ľubovoľne zvolenej knižnice.
4. Navrhnite si vlastnú entitu s min. 4 atribútmi napr. Čokoláda ☺, ktorú budeme vedieť pridávať do zoznamu čokolád. Vytvorte si json súbor s druhmi čokolád a preveďte túto kolekciu json dokumentov na java objekty typu čokoláda.