JSON

JSON (z anglického **JavaScript Object Notation**) je lehký, textový a člověkem čitelný formát pro výměnu a ukládání strukturovaných dat. Původně vznikl jako podmnožina jazyka JavaScript, dnes je však **nezávislý na jakémkoli programovacím jazyce** a podporován téměř všemi moderními technologiemi.

JSON se často používá pro:

- komunikaci mezi klientem a serverem (např. v REST API),
- konfigurační soubory,
- ukládání nestrukturovaných nebo polostrukturovaných dat (např. v NoSQL databázích jako MongoDB).

Syntaxe a datové typy

JSON podporuje následující základní datové typy:

- **String** – text uzavřený v **dvojitých uvozovkách**: `„Ahoj“`
- **Number** – celá nebo desetinná čísla (bez uvozovek): `42`, `-3.14`, `1.23e5`
- **Boolean** – logické hodnoty: `true`, `false`
- **Null** – prázdná hodnota: `null`
- **Object** – neuspořádaná množina dvojic klíč–hodnota (tzv. „mapa“ nebo „slovník“), uzavřená do složených závorek `{}`.
- **Array** – uspořádaný seznam hodnot, uzavřený do hranatých závorek `[]`.

 Důležité: V JSON **musí být vždy použity dvojité uvozovky** (`„`). Jednoduché uvozovky (`'`) nejsou platné.

Příklad JSON dokumentu

```
{
  "jmeno": "Jan",
  "prijmeni": "Novák",
  "vek": 32,
  "student": false,
  "email": null,
  "konicky": ["čtení", "běhání", "programování"],
  "adresa": {
    "ulice": "Náměstí Míru 123",
    "mesto": "Praha",
    "psc": "110 00",
    "zeme": "Česká republika"
  }
}
```

Pravidla syntaxe JSON

- Klíče (názvy vlastností) **musí být řetězce v dvojitých uvozovkách**.
- Hodnoty mohou být libovolného platného JSON typu.
- Čárky oddělují jednotlivé prvky (v poli) nebo dvojice klíč–hodnota (v objektu).
- **Na konci posledního prvku v objektu nebo poli nesmí být čárka** (tzv. „trailing comma“ – není v základním JSON povolena, i když některé knihovny ji tolerují).
- Komentáře **nejsou v oficiálním JSON standardu podporovány**.

Nesprávný JSON (běžné chyby):

```
{
  'jmeno': 'Jan',       // ❌ Jednoduché uvozovky
  vek: 32,              // ❌ Klíč bez uvozovek
  konicky: ["a", "b",], // ❌ Čárka na konci pole
  // komentář          // ❌ Komentáře nejsou povoleny
}
```

Výhody a nevýhody

Výhody	Nevýhody
Čitelnost – snadno čitelný jak pro lidi, tak pro stroje.	**Žádné komentáře** – nelze přidávat poznámky do konfiguračních souborů.
Lehký a kompaktní – menší objem než XML.	**Duplikace klíčů** – v objektu mohou být teoreticky duplicitní klíče (standard říká, že by měly být ignorovány – nebezpečné pro bezpečnost).
Široká podpora – všechny moderní jazyky mají vestavěnou podporu (např. `JSON.parse()` v JavaScriptu, `json` modul v Pythonu).	**Žádná kontrola schématu v základu** – bez validace může být struktura nekonzistentní (řeší se pomocí JSON Schema).
Hierarchická struktura – ideální pro vnořená data.	**Bez typové bezpečnosti** – např. číslo a řetězec se liší jen uvozovkami, což může vést k chybám.

Použití v praxi

- **Webová API**: Většina REST API vrací data ve formátu JSON (např. `Content-Type: application/json`).
- **Konfigurační soubory**: Např. `package.json` v Node.js, `tsconfig.json` v TypeScriptu.
- **NoSQL databáze**: MongoDB, CouchDB a Firebase ukládají data jako JSON dokumenty.
- **Ukládání dat v aplikacích**: Mnoho desktopových i mobilních aplikací používá JSON pro lokální ukládání nastavení nebo dat.

Příklad v JavaScriptu:

```
const data = JSON.parse('{"jmeno":"Jan","vek":32}');
console.log(data.jmeno); // "Jan"

const jsonStr = JSON.stringify(data);
```

```
// Vrátí: '{"jmeno":"Jan","vek":32}'
```

Příklad v Pythonu:

```python
import json

data = {"jmeno": "Jan", "vek": 32}
json_str = json.dumps(data, ensure_ascii=False)
# Výstup: {"jmeno": "Jan", "vek": 32}

obj = json.loads('{"jmeno": "Eva"}')
print(obj["jmeno"])  # "Eva"
```

Standardy a specifikace

- **RFC 8259** – oficiální specifikace JSON (nahrazuje původní RFC 7159).
- **MIME typ**: `application/json`
- **Přípona souboru**: `.json`
- **Kódování**: JSON musí být v kódování **UTF-8**, **UTF-16** nebo **UTF-32**, přičemž **UTF-8 je doporučeno**.

Rozšíření a nástroje

- **JSON5** – rozšíření JSON s podporou komentářů, jednoduchých uvozovek, trailing čárek atd. (není kompatibilní s čistým JSON).
- **JSON Schema** – jazyk pro popis a validaci struktury JSON dat.
- **jq** – příkazový nástroj pro filtrování a transformaci JSON v terminálu.
- **YAML** – alternativní formát, který je člověkem ještě lépe čitelný, ale složitější.

Bezpečnostní rizika

- **JSON injection** – pokud se data z JSON nevalidují, mohou být použita k útokům (např. XSS).
- **Prototype pollution** – v JavaScriptu může škodlivý JSON manipulovat s vlastnostmi objektů (zejména při použití `JSON.parse()` s nebezpečnými knihovnami).
- **Únik dat** – konfigurační soubory `.json` často obsahují citlivé údaje (API klíče, hesla) – nikdy je neukládej do veřejných repozitářů!

Související pojmy

- XML
- CSV
- REST API
- JSON Schema
- NoSQL
- MIME typy

Externí odkazy

- Oficiální web: https://www.json.org/json-cs.html
- RFC 8259: https://datatracker.ietf.org/doc/html/rfc8259
- JSON Schema: https://json-schema.org
- JSON Validator: https://jsonlint.com

Viz také

- API
- Datové formáty
- Kódování znaků
- Konfigurační soubory