

REST API (Representational State Transfer)

REST API je rozhraní, které využívá principy architektury REST pro výměnu dat mezi klientem (např. webový prohlížeč) a serverem. Je založeno na bezstavovosti a využívá standardní metody protokolu HTTP.

1. Základní principy RESTu

Aby bylo rozhraní považováno za „RESTful“, mělo by splňovat několik pravidel:

- **Klient-Server:** Oddělení uživatelského rozhraní od datového úložiště.
- **Bezstavovost (Statelessness):** Každý požadavek od klienta musí obsahovat všechny informace potřebné k jeho zpracování. Server si mezi požadavky nepamatuje žádný „stav“ (kontext) klienta.
- **Jednotné rozhraní:** Komunikace probíhá pomocí standardních URL adres (zdrojů) a HTTP metod.
- **Cacheovatelnost:** Odpovědi serveru by měly definovat, zda je možné je uložit do mezipaměti (cache), aby se ušetřil přenos dat.

—

2. HTTP Metody (Slovesa)

REST API využívá HTTP metody k definování akce, která se má se zdrojem provést:

Metoda	Akce	Význam
GET	Čtení	Získání dat o zdroji (např. seznam uživatelů).
POST	Vytvoření	Vytvoření nového zdroje (např. registrace uživatele).
PUT	Aktualizace	Nahrazení stávajícího zdroje novými daty.
PATCH	Částečná aktualizace	Úprava pouze části zdroje (např. změna hesla).
DELETE	Smazání	Odstranění zdroje.

—

3. Zdroje a URL struktura

V RESTu je vše považováno za **zdroj** (resource), ke kterému se přistupuje pomocí unikátní adresy (URI). Názvy v URL by měly být podstatná jména v množném čísle.

Příklady:

- `GET /users` – Získá seznam všech uživatelů.
- `GET /users/42` – Získá detail uživatele s ID 42.
- `POST /users` – Vytvoří nového uživatele (data jsou v těle požadavku).

4. Formát dat (JSON)

Ačkoliv REST může teoreticky používat i XML nebo text, naprostým standardem je dnes **JSON** (JavaScript Object Notation). Je lehký, dobře čitelný pro lidi a snadno zpracovatelný pro stroje.

```
// Příklad odpovědi ve formátu JSON
{
  "id": 42,
  "jmeno": "Jan Novák",
  "email": "jan.novak@example.com"
}
```

5. Stavové kódy HTTP

Server informuje klienta o výsledku požadavku pomocí číselných kódů:

- **2xx (Success):** Např. **200 OK** (vše v pořádku) nebo **201 Created** (úspěšně vytvořeno).
- **4xx (Client Error):** Např. **400 Bad Request** (chyba v datech od klienta) nebo **404 Not Found** (zdroj neexistuje).
- **5xx (Server Error):** Např. **500 Internal Server Error** (chyba na straně serveru).

Související články:

- [Protokol HTTP hloubkově](#)
- [ASP.NET Core: Tvorba Web API](#)
- [Formát JSON: Specifikace](#)

Tagy: programming web api rest json http architecture

From:
<http://serviceit.cz/> - **IT ENCYKLOPEDIE**

Permanent link:
http://serviceit.cz/doku.php?id=it:sw:rest_api

Last update: **2026/01/02 19:51**

