# **json-server**

## Install

npm install json-server

## Usage

Create a db.json or db.json5 file

{

"posts": [

{ "id": "1", "title": "a title", "views": 100 },

{ "id": "2", "title": "another title", "views": 200 }

],

"comments": [

{ "id": "1", "text": "a comment about post 1", "postId": "1" },

{ "id": "2", "text": "another comment about post 1", "postId": "1" }

],

"profile": {

"name": "typicode"

}

}

View db.json5 example

Pass it to JSON Server CLI

$ npx json-server db.json

Get a REST API

$ curl http://localhost:3000/posts/1

{

"id": "1",

"title": "a title"

}

Run json-server --help for a list of options

**Routes**

Based on the example db.json, you'll get the following routes:

GET /posts

GET /posts/:id

POST /posts

PUT /posts/:id

PATCH /posts/:id

DELETE /posts/:id

# Same for comments

GET /profile

PUT /profile

PATCH /profile

**Params**

**Conditions**

* → ==
* lt → <
* lte → <=
* gt → >
* gte → >=
* ne → !=

GET /posts?views\_gt=9000

**Range**

* start
* end
* limit

GET /posts?\_start=10&\_end=20

GET /posts?\_start=10&\_limit=10

**Paginate**

* page
* per\_page (default = 10)

GET /posts?\_page=1&\_per\_page=25

**Sort**

* \_sort=f1,f2

GET /posts?\_sort=id,-views

**Nested and array fields**

* x.y.z...
* x.y.z[i]...

GET /foo?a.b=bar

GET /foo?x.y\_lt=100

GET /foo?arr[0]=bar

**Embed**

GET /posts?\_embed=comments

GET /comments?\_embed=post

**Delete**

DELETE /posts/1

DELETE /posts/1?\_dependent=comments

**Serving static files**

If you create a ./public directory, JSON Serve will serve its content in addition to the REST API.

You can also add custom directories using -s/--static option.

json-server -s ./static

json-server -s ./static -s ./node\_modules

**Notable differences with v0.17**

* id is always a string and will be generated for you if missing
* use \_per\_page with \_page instead of \_limitfor pagination
* use Chrome's Network tab > throtling to delay requests instead of --delay CLI option