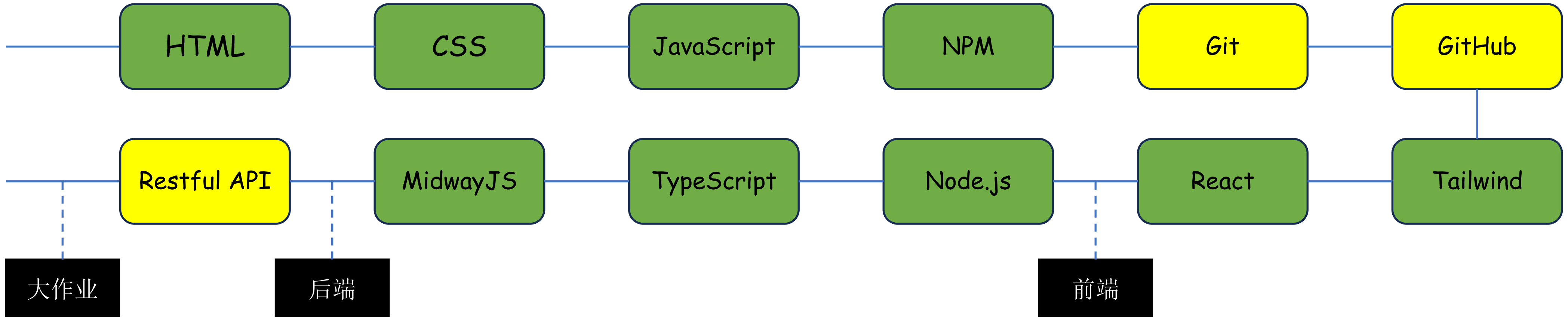
Web 前后端通信及测试技术

章许帆


HTML
CSS
JavaScript
NPM
Git
GitHub
Tailwind
React
Node.js
TypeScript
MidwayJS
Restful API
大作业
后端
前端

Web 前后端交互技术及测试

HTTP

REST 风格API

WebSocket

前后端交互

测试指标

单元测试

接口测试

数据校验

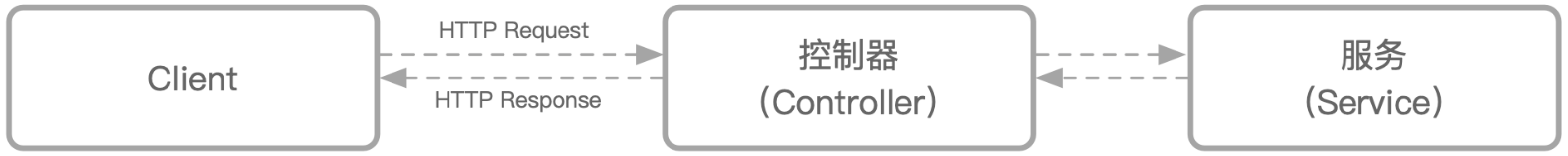
测试

JEST

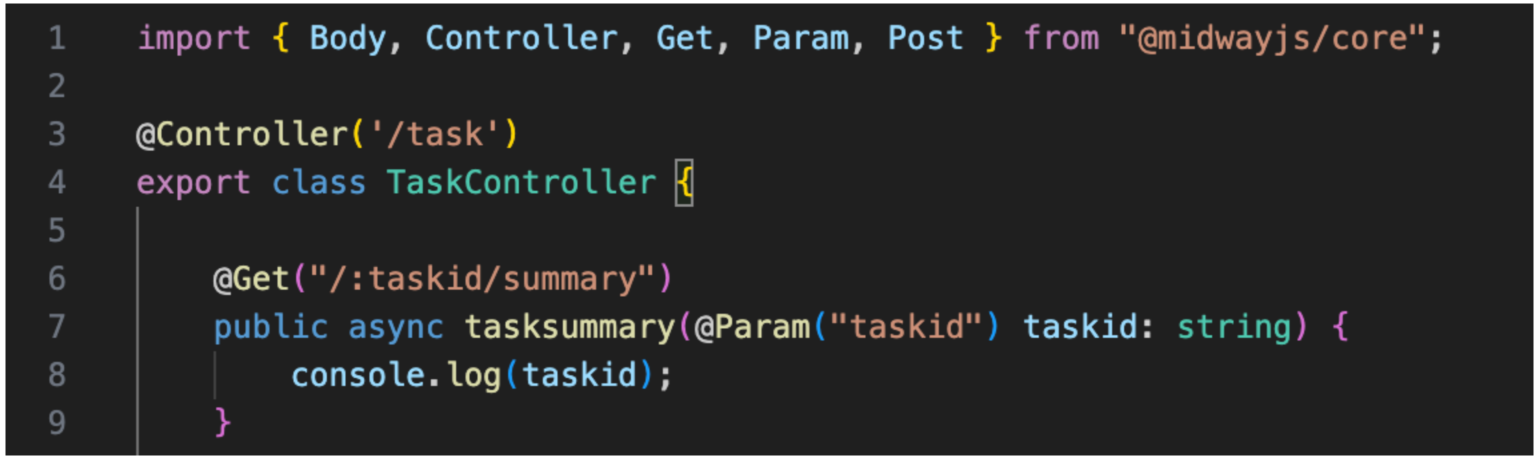
ESLint

实验: 编写 Web App 测试

后端代码结构


Client
HTTP Request
HTTP Response
控制器
(Controller)
服务
(Service)

HTTP Params - Router Params

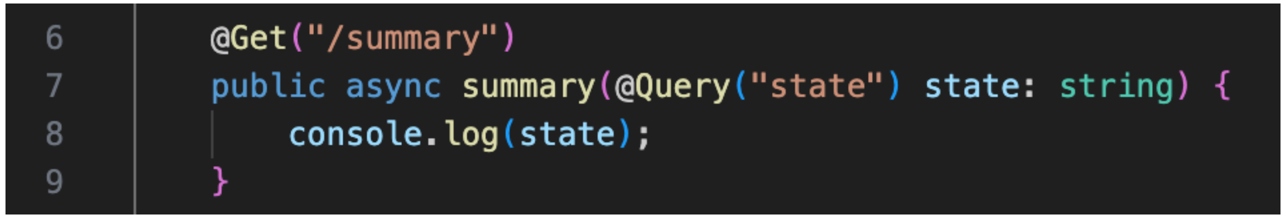


```
import { Body, Controller, Get, Param, Post } from "@midwayjs/core";

@Controller('/task')
export class TaskController {

    @Get("/:taskid/summary")
    public async tasksummary(@Param("taskid") taskid: string) {
        console.log(taskid);
    }
```

Router Params

GET http://localhost:7001/task/233/summary

HTTP Params - Query String

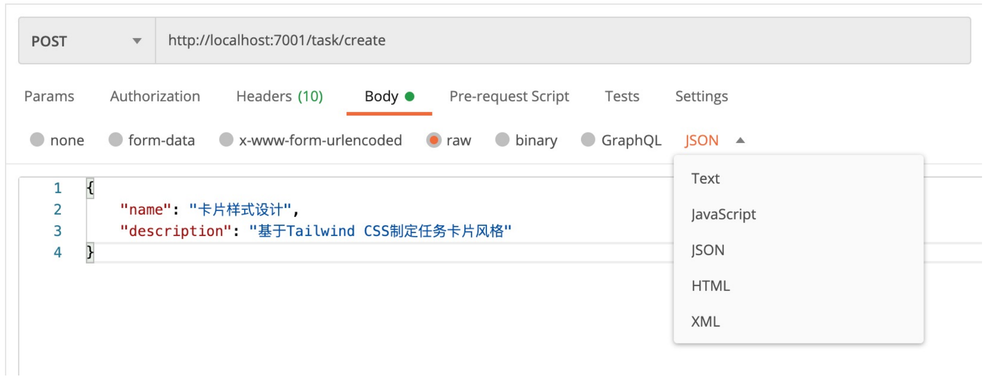


```
@Get("/summary")
public async summary(@Query("state") state: string) {
    console.log(state);
}
```

Query String

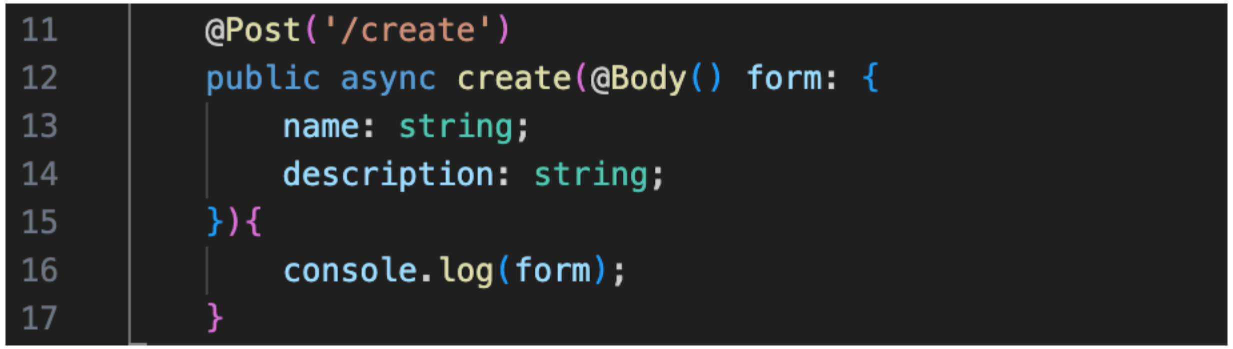
GET http://localhost:7001/task/summary?state=finished

HTTP Params - Body Params



http://localhost:7001/task/create

```
{
    "name": "卡片样式设计",
    "description": "基于Tailwind CSS制定任务卡片风格"
}
```



```
    @Post('/create')
    public async create(@Body() form: {
        name: string;
        description: string;
    }){
        console.log(form);
    }
```

HTTP Body

POST http://localhost:7001/task/create

HTTP RESTful

资源导向

无状态

容易扩展

HTTP POST /task/create

HTTP GET   /task/{taskid}/summary   *<u>HTTP GET   /task/summary?taskid=xxx</u>*

HTTP GET   /interests/overview

HTTP POST /threads/{threadid}/comment   *<u>HTTP POST /threads/comment?threadid=xxx</u>*

axios 前后端通信实现

基于Promise的 HTTP 客户端

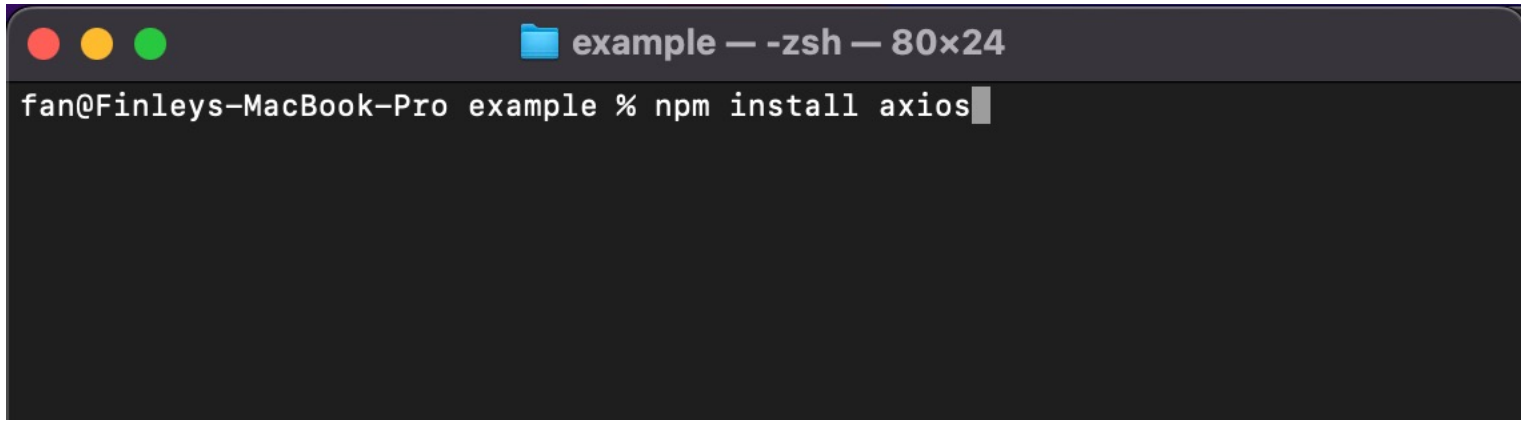
Koa

前端项目

XMLHttpRequest

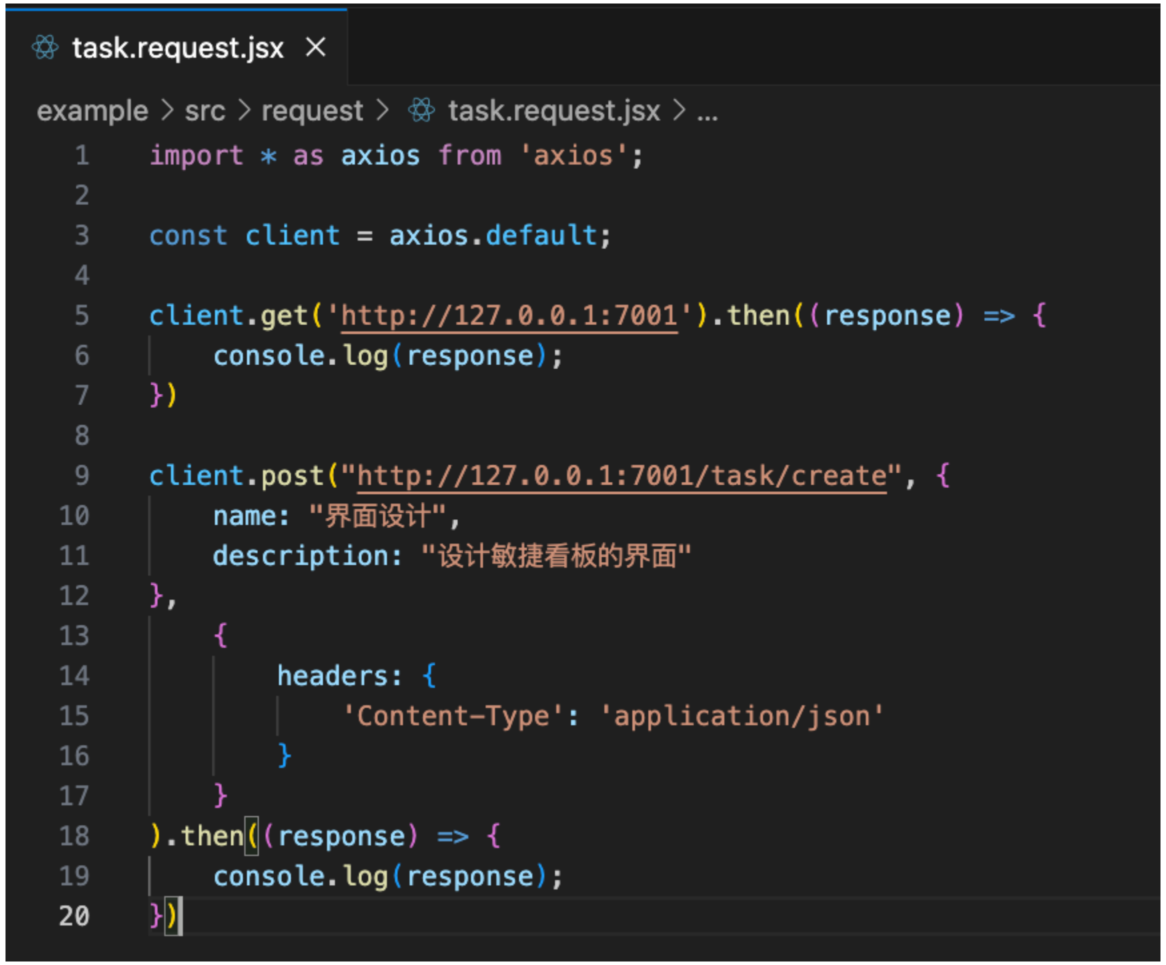
后端项目

http



```
fan@Finleys-MacBook-Pro example % npm install axios
```

axios request



```
import * as axios from 'axios';

const client = axios.default;

client.get('http://127.0.0.1:7001').then((response) => {
    console.log(response);
})

client.post("http://127.0.0.1:7001/task/create", {
    name: "界面设计",
    description: "设计敏捷看板的界面"
},
    {
        headers: {
            'Content-Type': 'application/json'
        }
    }
).then((response) => {
    console.log(response);
})
```

哪个请求先返回结果?

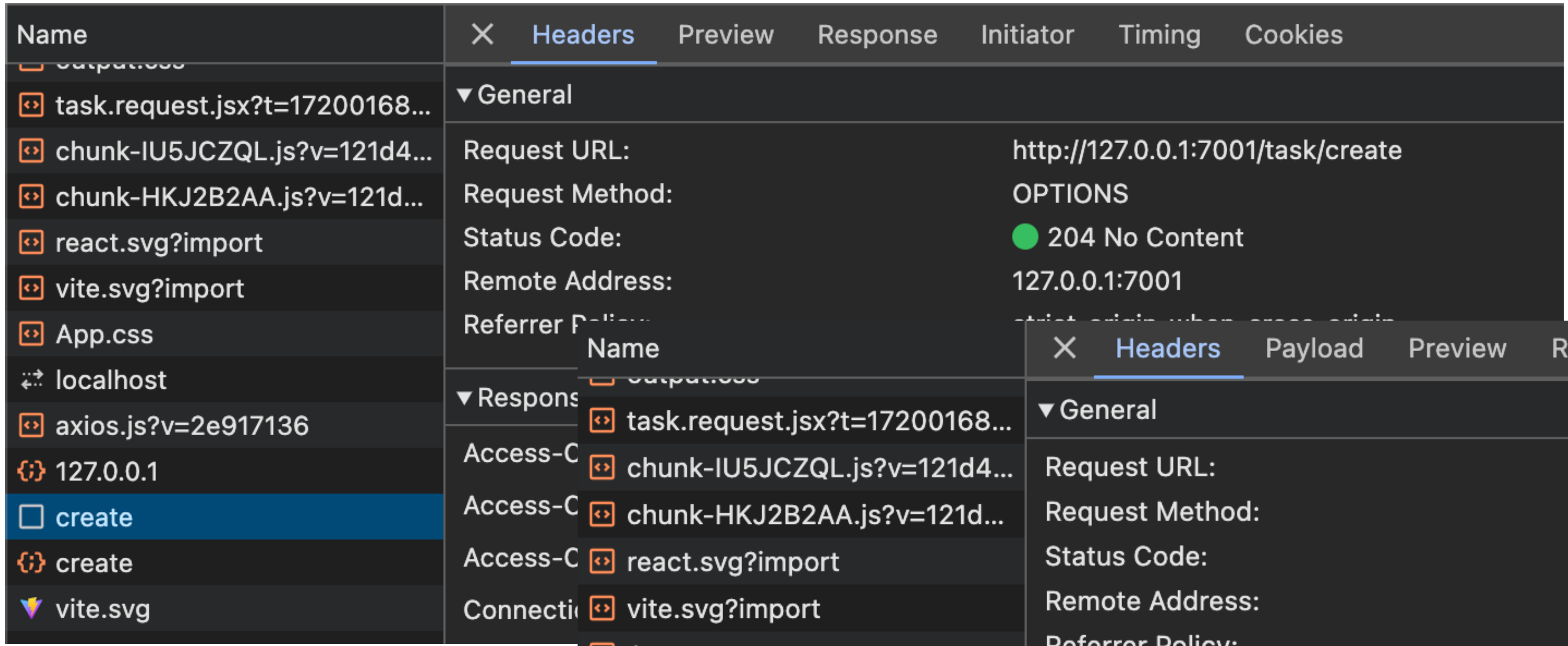
跨域CORS

Access to XMLHttpRequest at 'http://127.0.0.1:7001/task/create' from origin 'http://localhost:5173' has been blocked by CORS policy: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource.

https://midwayjs.org/docs/extensions/cross_domain

```
// src/config/config.default.ts
export default {
  // ...
  cors: {
    origin: '*',
  },
}
```

HTTP Options



Name

task.request.jsx?t=17200168...
chunk-IU5JCZQL.js?v=121d4...
chunk-HKJ2B2AA.js?v=121d...
react.svg?import
vite.svg?import
App.css
localhost
axios.js?v=2e917136
127.0.0.1
create
create
vite.svg

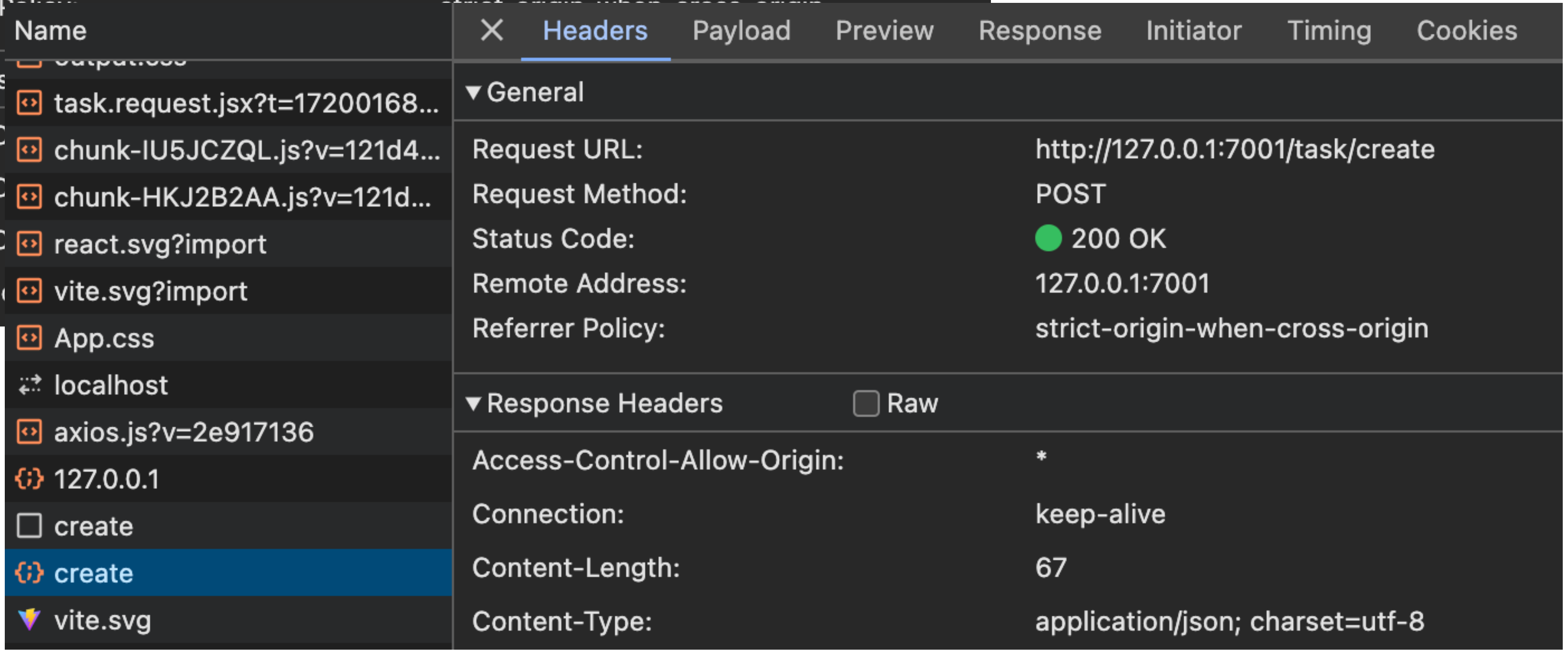
Headers | Preview | Response | Initiator | Timing | Cookies

▼General

Request URL:	http://127.0.0.1:7001/task/create
Request Method:	OPTIONS
Status Code:	● 204 No Content
Remote Address:	127.0.0.1:7001

▼Respons

Access-C
Access-C
Access-C
Connecti



Name

task.request.jsx?t=17200168...
chunk-IU5JCZQL.js?v=121d4...
chunk-HKJ2B2AA.js?v=121d...
react.svg?import
vite.svg?import
App.css
localhost
axios.js?v=2e917136
127.0.0.1
create
create
vite.svg

Headers | Payload | Preview | Response | Initiator | Timing | Cookies

▼General

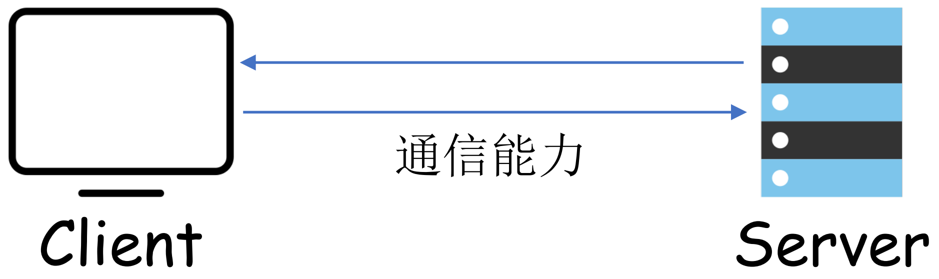
Request URL:	http://127.0.0.1:7001/task/create
Request Method:	POST
Status Code:	● 200 OK
Remote Address:	127.0.0.1:7001
Referrer Policy:	strict-origin-when-cross-origin

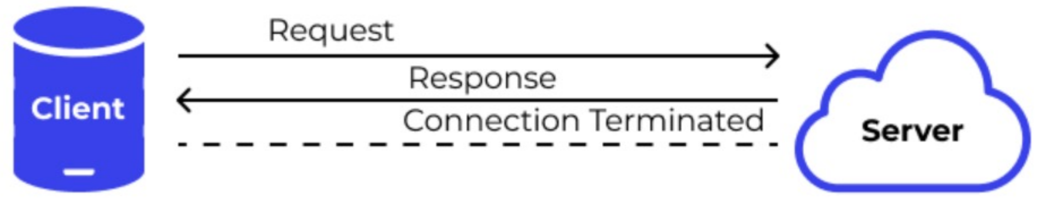
▼Response Headers ☐ Raw

Access-Control-Allow-Origin:	*
Connection:	keep-alive
Content-Length:	67
Content-Type:	application/json; charset=utf-8

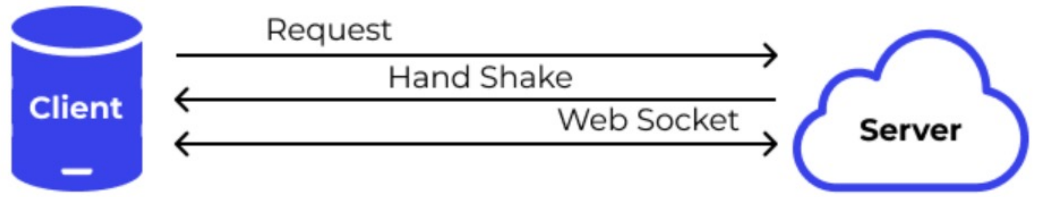
Client端如何实时获取数据?

HTTP VS. WebSocket



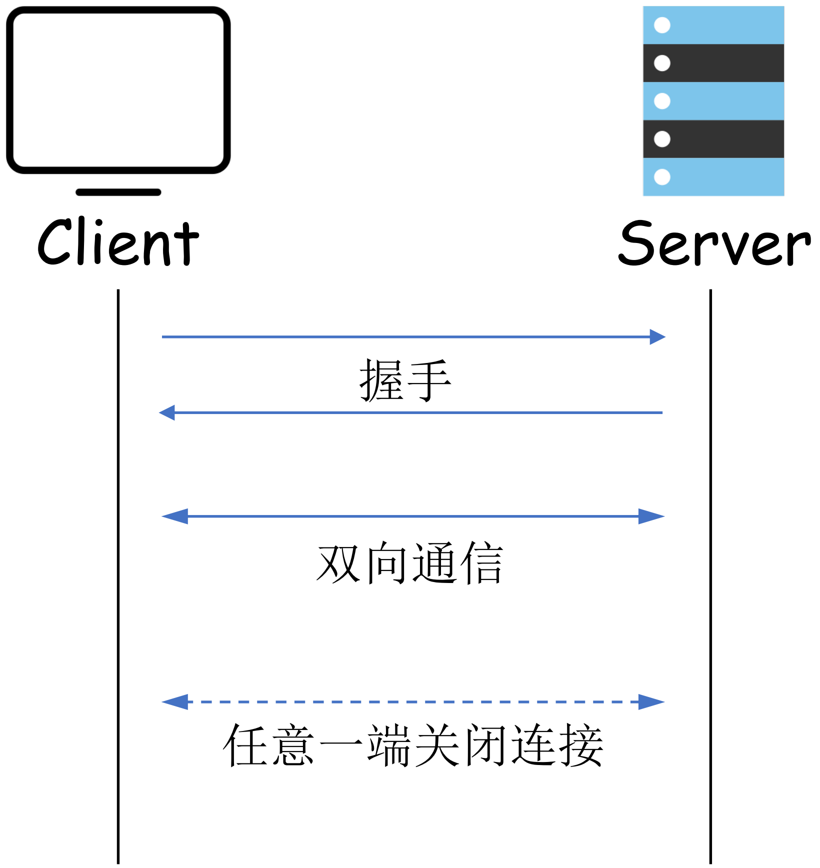


HTTP

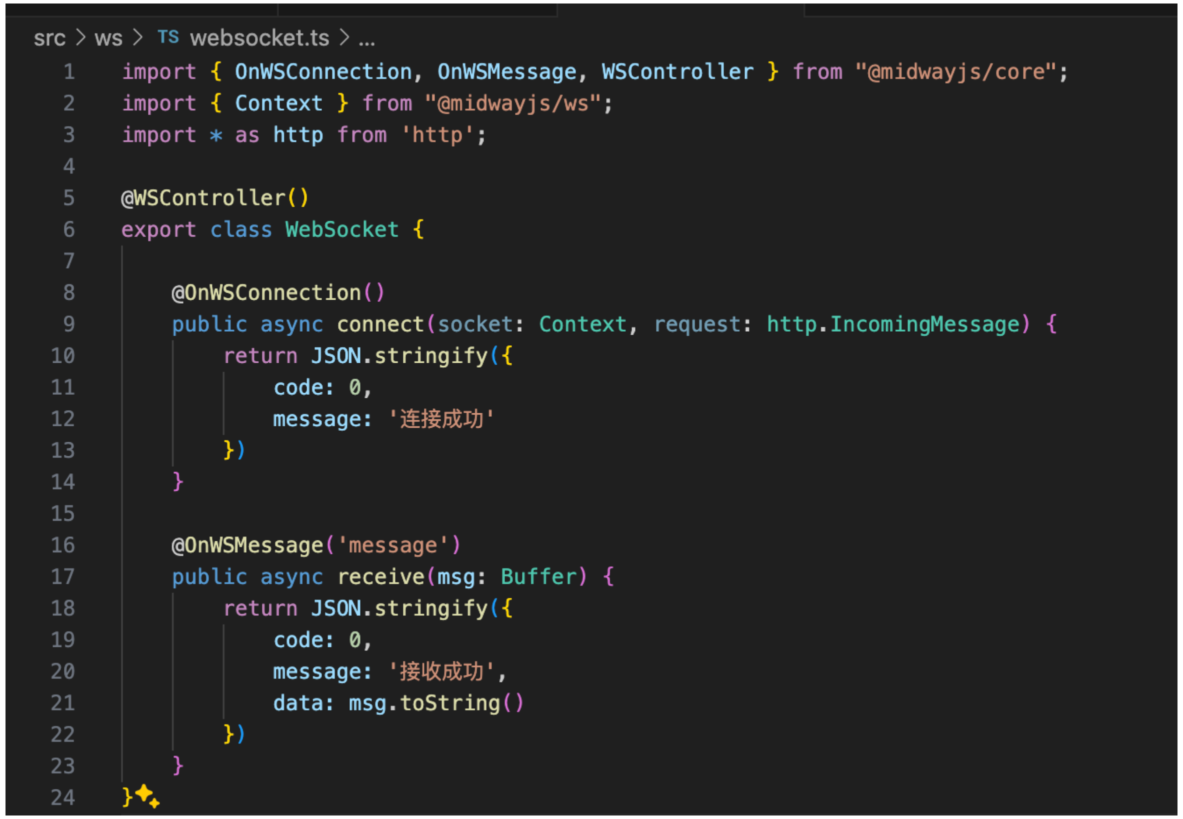


WebSocket

WebSocket

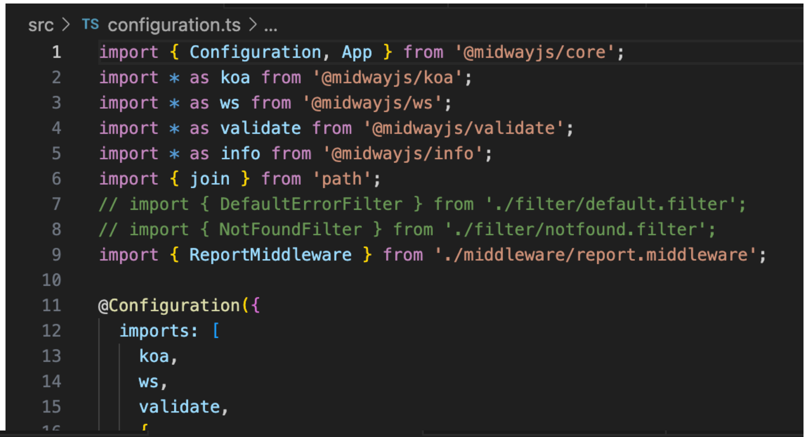


HTTP + WebSocket



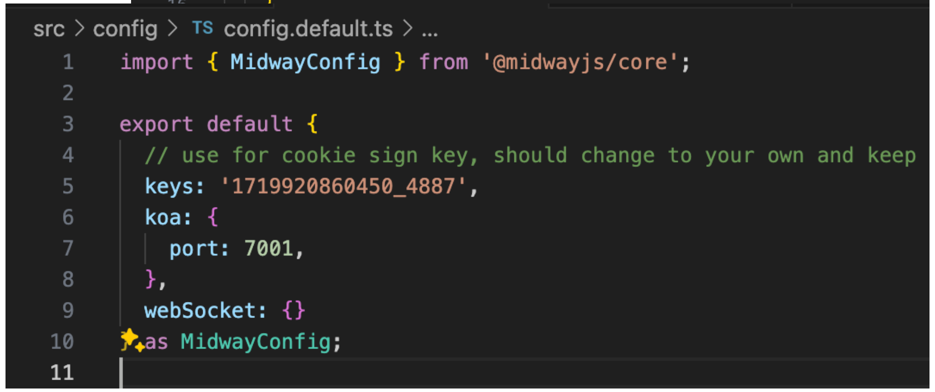
src > ws > TS websocket.ts > ...

```ts
import { OnWSConnection, OnWSMessage, WSController } from "@midwayjs/core";
import { Context } from "@midwayjs/ws";
import * as http from 'http';

@WSController()
export class WebSocket {

    @OnWSConnection()
    public async connect(socket: Context, request: http.IncomingMessage) {
        return JSON.stringify({
            code: 0,
            message: '连接成功'
        })
    }

    @OnWSMessage('message')
    public async receive(msg: Buffer) {
        return JSON.stringify({
            code: 0,
            message: '接收成功',
            data: msg.toString()
        })
    }
}
```



src > TS configuration.ts > ...

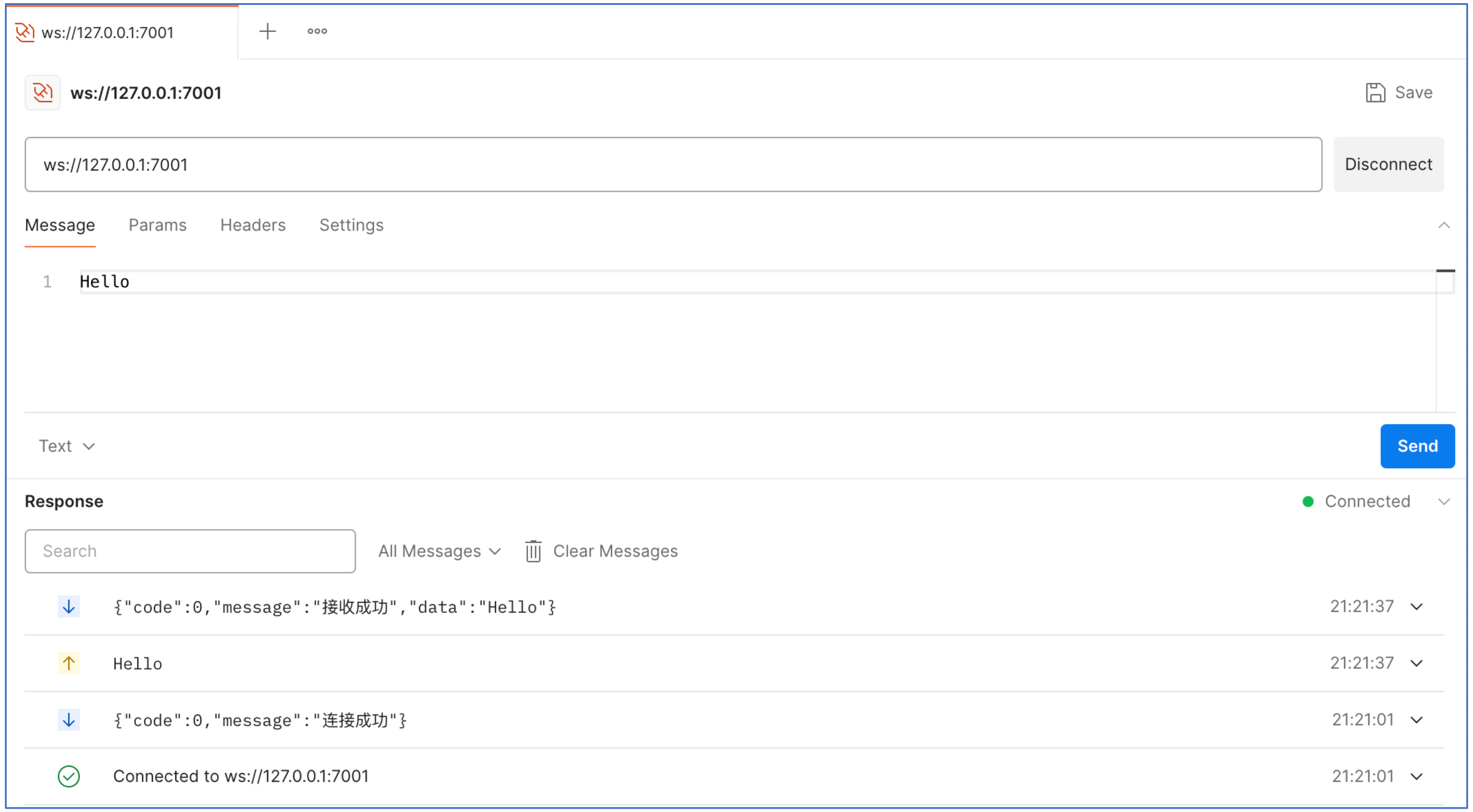
```ts
import { Configuration, App } from '@midwayjs/core';
import * as koa from '@midwayjs/koa';
import * as ws from '@midwayjs/ws';
import * as validate from '@midwayjs/validate';
import * as info from '@midwayjs/info';
import { join } from 'path';
// import { DefaultErrorFilter } from './filter/default.filter';
// import { NotFoundFilter } from './filter/notfound.filter';
import { ReportMiddleware } from './middleware/report.middleware';

@Configuration({
  imports: [
    koa,
    ws,
    validate,
```



src > config > TS config.default.ts > ...

```ts
import { MidwayConfig } from '@midwayjs/core';

export default {
  // use for cookie sign key, should change to your own and keep
  keys: '1719920860450_4887',
  koa: {
    port: 7001,
  },
  webSocket: {}
} as MidwayConfig;

```

WebSocket 通信

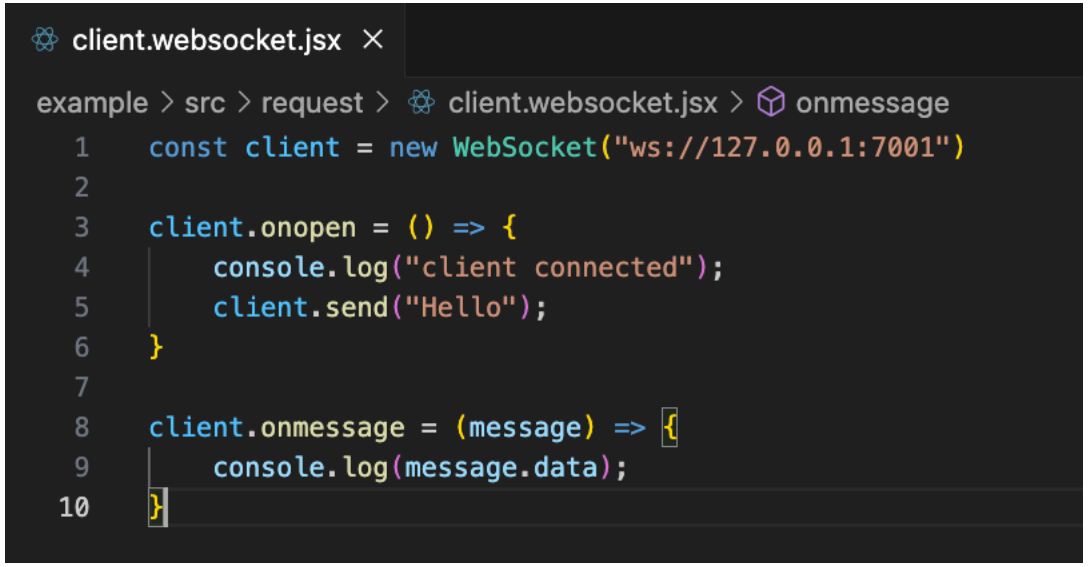


ws://127.0.0.1:7001

```
Hello
```

```
{"code":0,"message":"接收成功","data":"Hello"}   21:21:37
Hello   21:21:37
{"code":0,"message":"连接成功"}   21:21:01
Connected to ws://127.0.0.1:7001   21:21:01
```

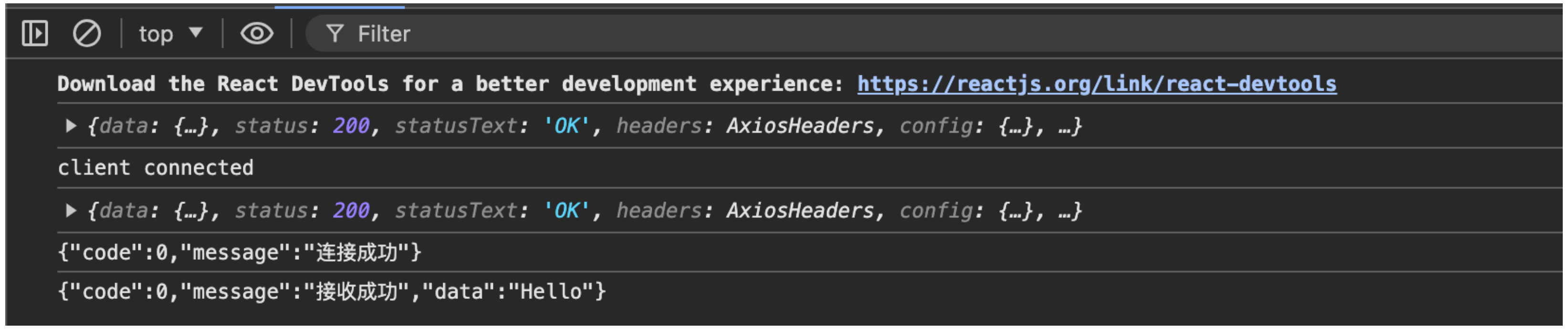
WebSocket 通信



client.websocket.jsx

example > src > request > client.websocket.jsx > onmessage

```jsx
const client = new WebSocket("ws://127.0.0.1:7001")

client.onopen = () => {
    console.log("client connected");
    client.send("Hello");
}

client.onmessage = (message) => {
    console.log(message.data);
}
```



```
Download the React DevTools for a better development experience: https://reactjs.org/link/react-devtools
▶ {data: {…}, status: 200, statusText: 'OK', headers: AxiosHeaders, config: {…}, …}
client connected
▶ {data: {…}, status: 200, statusText: 'OK', headers: AxiosHeaders, config: {…}, …}
{"code":0,"message":"连接成功"}
{"code":0,"message":"接收成功","data":"Hello"}
```

参数校验

```
@Post('/create')
public async create(@Body() form: {
    name: string;
    description: string;
}) {
     if (form.name !== undefined && form.description !== undefined) {
     console.log(form);
   } else {
     console.error("Invalid form data");
   }
}
```

业务逻辑代码中会充斥大量的数据校验代码

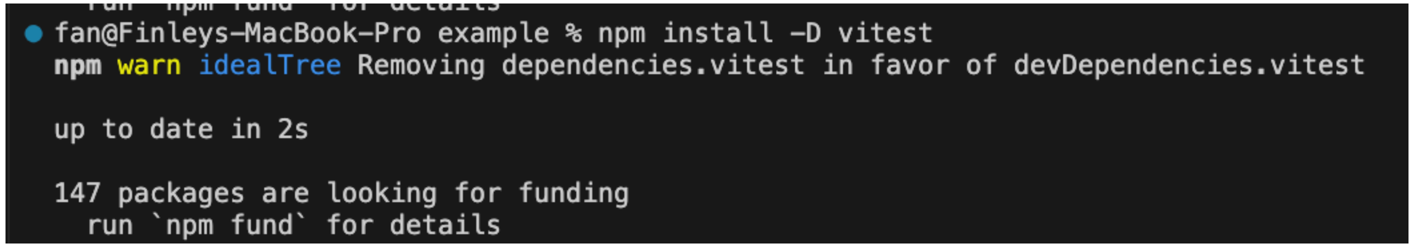
@midwayjs/validate

```
import { Rule, RuleType } from "@midwayjs/validate";

export class TaskDto {
    @Rule(RuleType.string().required())
    name: string;

    @Rule(RuleType.string().required())
    description: string;
}
```

我们通常会怎么描述软件的问题

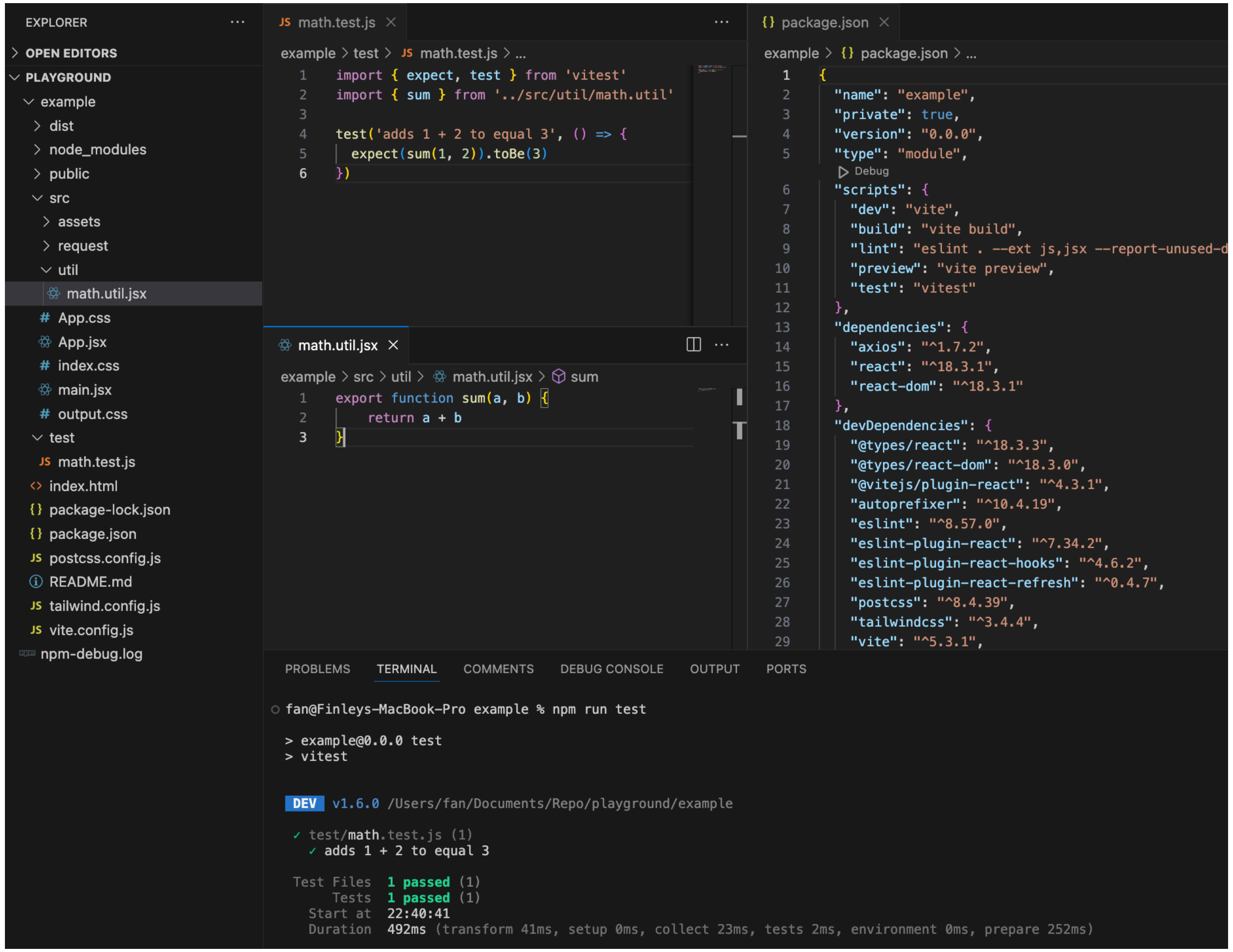
执行某个操作报错	功能的正确性
界面卡顿，点了没反应	功能、性能
功能不好用	易用性

前端测试 Vitest



```
● fan@Finleys-MacBook-Pro example % npm install -D vitest
  npm warn idealTree Removing dependencies.vitest in favor of devDependencies.vitest

  up to date in 2s

  147 packages are looking for funding
    run `npm fund` for details
```

前端测试 Vitest



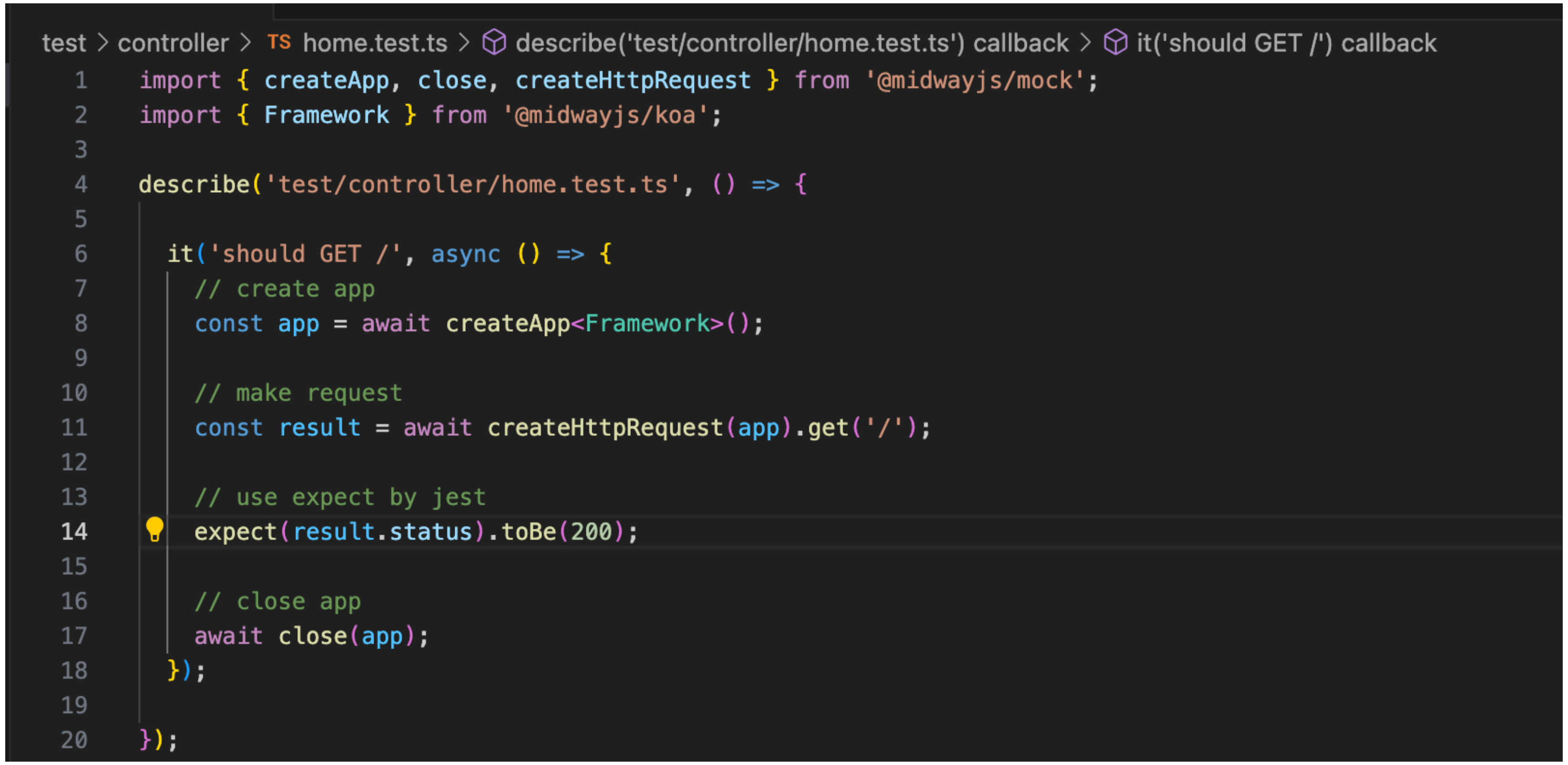
```js
import { expect, test } from 'vitest'
import { sum } from '../src/util/math.util'

test('adds 1 + 2 to equal 3', () => {
  expect(sum(1, 2)).toBe(3)
})
```

```jsx
export function sum(a, b) {
    return a + b
}
```

```json
{
  "name": "example",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "lint": "eslint . --ext js,jsx --report-unused-d
    "preview": "vite preview",
    "test": "vitest"
  },
  "dependencies": {
    "axios": "^1.7.2",
    "react": "^18.3.1",
    "react-dom": "^18.3.1"
  },
  "devDependencies": {
    "@types/react": "^18.3.3",
    "@types/react-dom": "^18.3.0",
    "@vitejs/plugin-react": "^4.3.1",
    "autoprefixer": "^10.4.19",
    "eslint": "^8.57.0",
    "eslint-plugin-react": "^7.34.2",
    "eslint-plugin-react-hooks": "^4.6.2",
    "eslint-plugin-react-refresh": "^0.4.7",
    "postcss": "^8.4.39",
    "tailwindcss": "^3.4.4",
    "vite": "^5.3.1",
```

```
○ fan@Finleys-MacBook-Pro example % npm run test

> example@0.0.0 test
> vitest


 DEV  v1.6.0 /Users/fan/Documents/Repo/playground/example

 ✓ test/math.test.js (1)
   ✓ adds 1 + 2 to equal 3

 Test Files  1 passed (1)
      Tests  1 passed (1)
   Start at  22:40:41
   Duration  492ms (transform 41ms, setup 0ms, collect 23ms, tests 2ms, environment 0ms, prepare 252ms)
```

后端测试 Jest



test > controller > TS home.test.ts > describe('test/controller/home.test.ts') callback > it('should GET /') callback

```ts
import { createApp, close, createHttpRequest } from '@midwayjs/mock';
import { Framework } from '@midwayjs/koa';

describe('test/controller/home.test.ts', () => {

  it('should GET /', async () => {
    // create app
    const app = await createApp<Framework>();

    // make request
    const result = await createHttpRequest(app).get('/');

    // use expect by jest
    expect(result.status).toBe(200);

    // close app
    await close(app);
  });

});
```

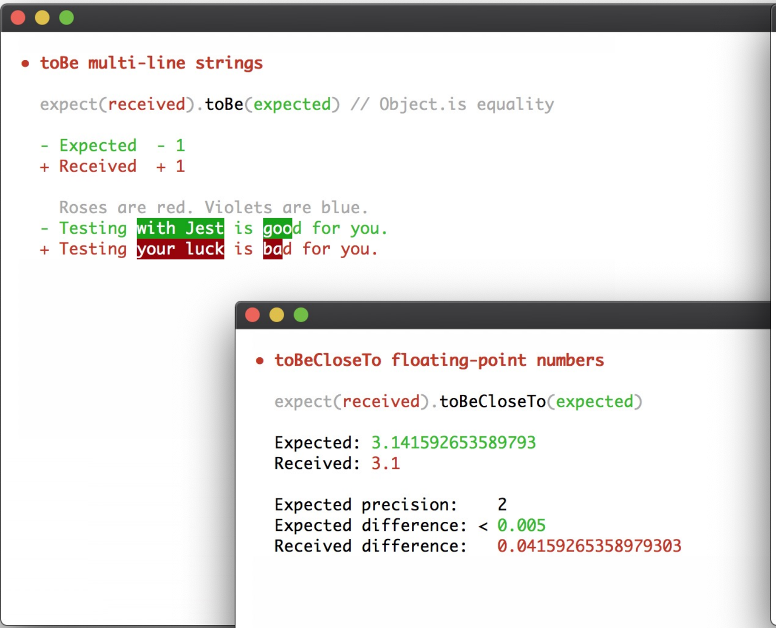
https://jestjs.io/docs/expect

断言 Assert

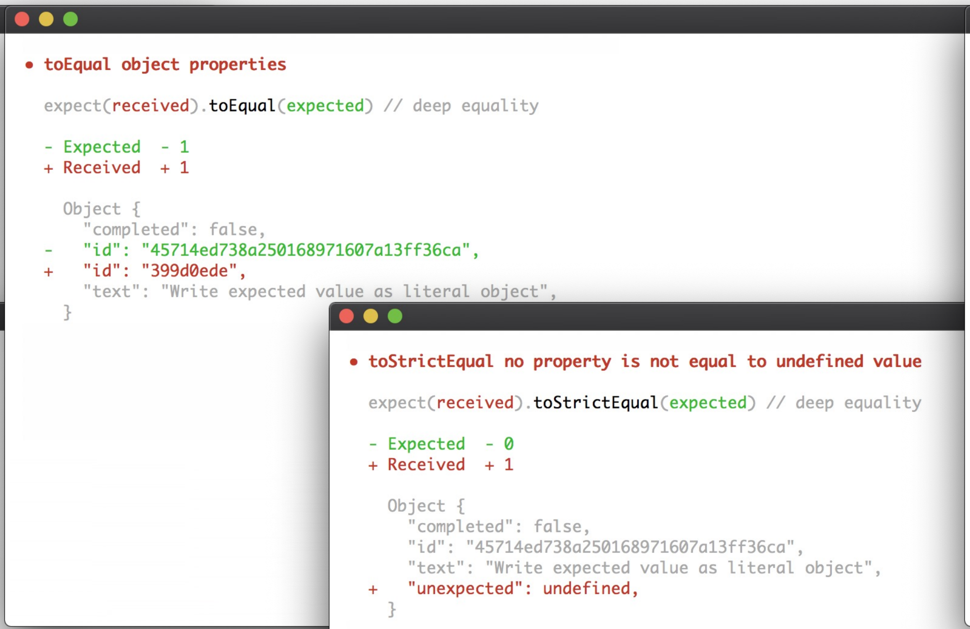
```js
expect(result.status).toBe(200);                   // 值是否等于某个值，引用相等
expect(result.status).not.toBe(200);
expect(result).toEqual('hello');                   // 简单匹配，对象属性相同也为 true
expect(result).toStrictEqual('hello');             // 严格匹配
expect(['lime', 'apple']).toContain('lime');  // 判断是否在数组中
```

https://jestjs.io/docs/en/expect

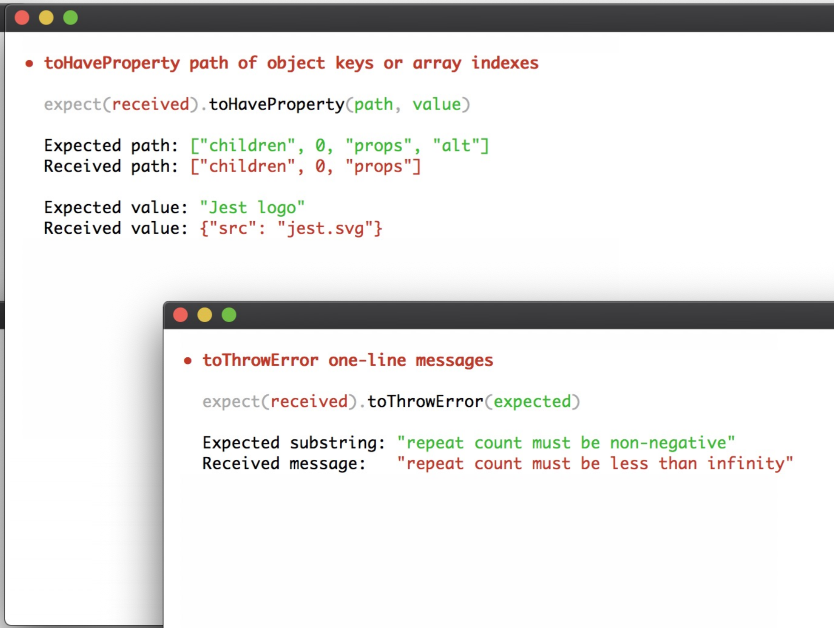
Examples



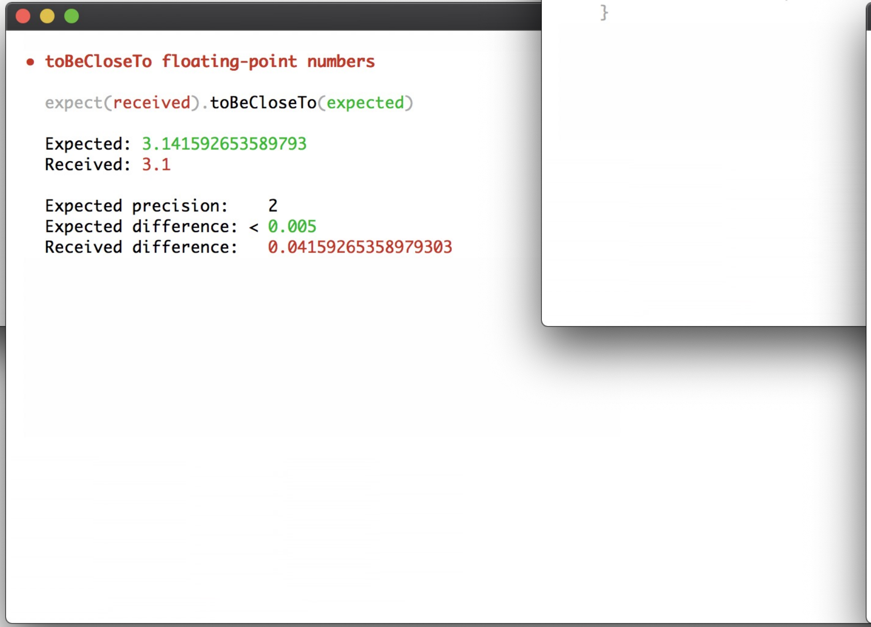
```
• toBe multi-line strings

  expect(received).toBe(expected) // Object.is equality

  - Expected  - 1
  + Received  + 1

    Roses are red. Violets are blue.
  - Testing with Jest is good for you.
  + Testing your luck is bad for you.
```



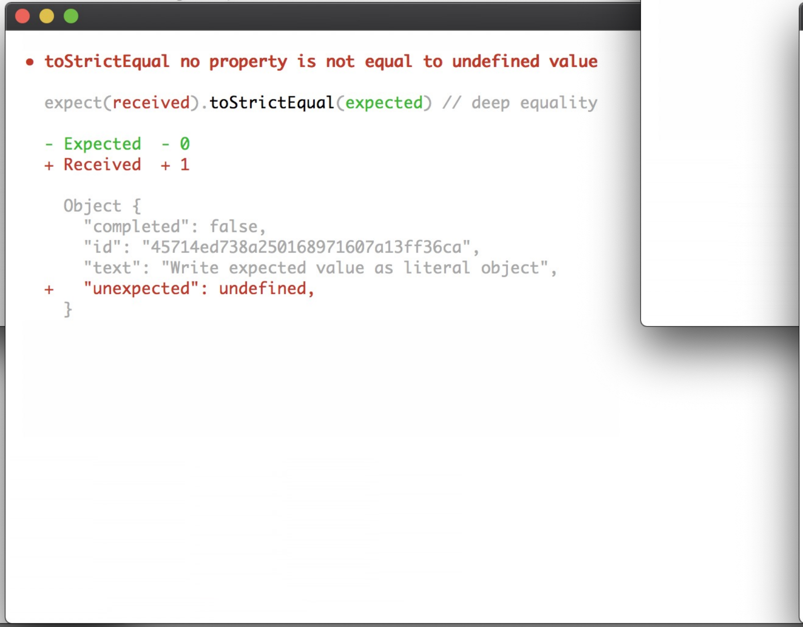
```
• toEqual object properties

  expect(received).toEqual(expected) // deep equality

  - Expected  - 1
  + Received  + 1

    Object {
      "completed": false,
  -   "id": "45714ed738a250168971607a13ff36ca",
  +   "id": "399d0ede",
      "text": "Write expected value as literal object",
    }
```



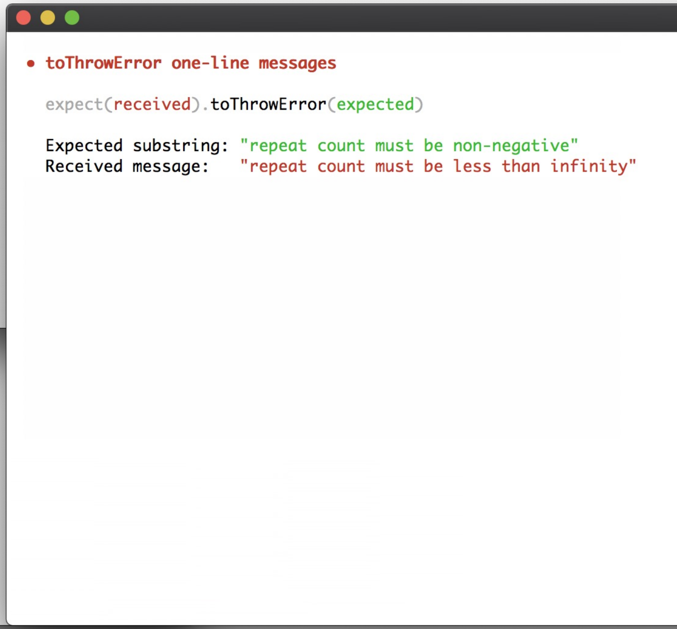
```
• toHaveProperty path of object keys or array indexes

  expect(received).toHaveProperty(path, value)

  Expected path: ["children", 0, "props", "alt"]
  Received path: ["children", 0, "props"]

  Expected value: "Jest logo"
  Received value: {"src": "jest.svg"}
```



```
• toBeCloseTo floating-point numbers

  expect(received).toBeCloseTo(expected)

  Expected: 3.141592653589793
  Received: 3.1

  Expected precision:    2
  Expected difference: < 0.005
  Received difference:   0.041592653589793
```

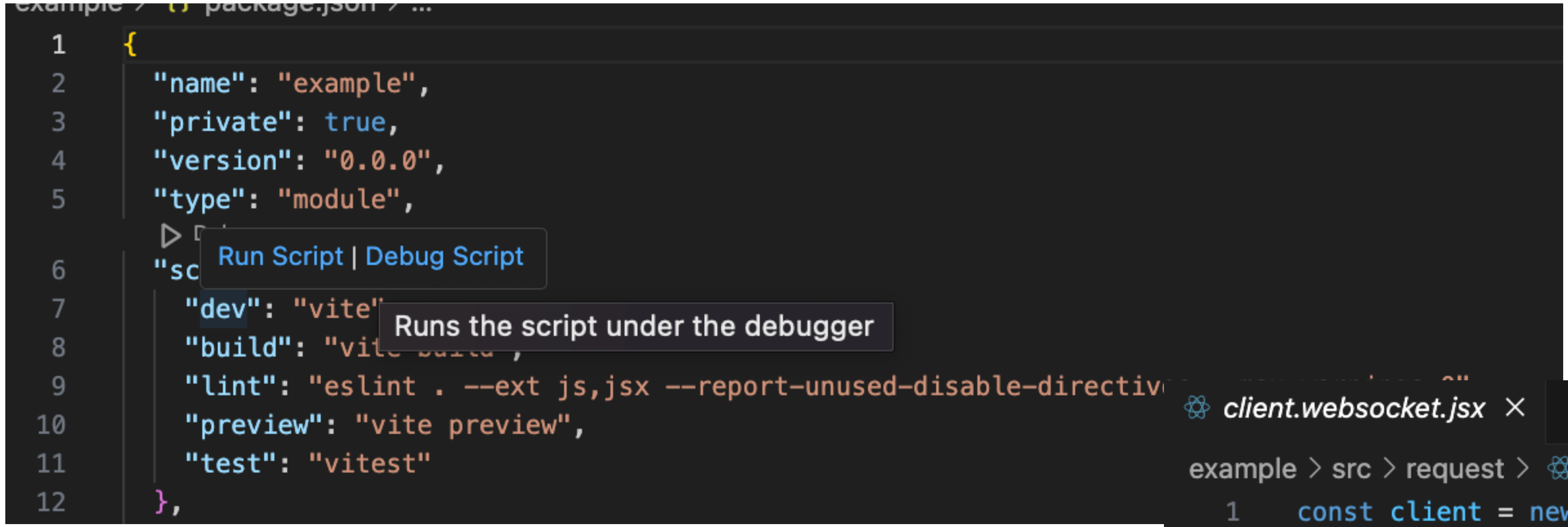


```
• toStrictEqual no property is not equal to undefined value

  expect(received).toStrictEqual(expected) // deep equality

  - Expected  - 0
  + Received  + 1

    Object {
      "completed": false,
      "id": "45714ed738a250168971607a13ff36ca",
      "text": "Write expected value as literal object",
  +   "unexpected": undefined,
    }
```



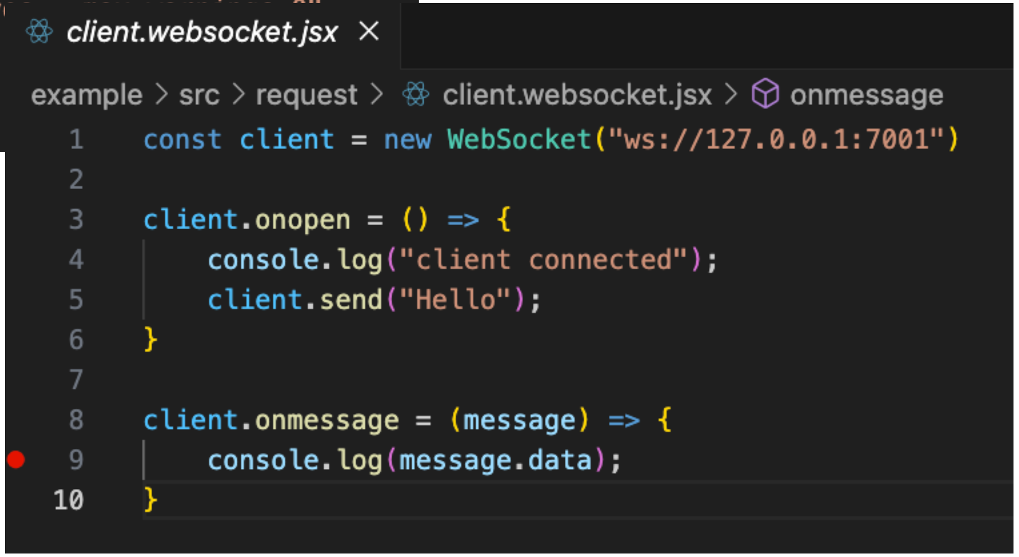
```
• toThrowError one-line messages

  expect(received).toThrowError(expected)

  Expected substring: "repeat count must be non-negative"
  Received message:   "repeat count must be less than infinity"
```

断点调试



example > {} package.json > ...
Run Script | Debug Script
Runs the script under the debugger

```json
{
  "name": "example",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  "sc
    "dev": "vite"
    "build": "vit
    "lint": "eslint . --ext js,jsx --report-unused-disable-directiv
    "preview": "vite preview",
    "test": "vitest"
  },
```



```jsx
const client = new WebSocket("ws://127.0.0.1:7001")

client.onopen = () => {
    console.log("client connected");
    client.send("Hello");
}

client.onmessage = (message) => {
    console.log(message.data);
}
```

谢　谢