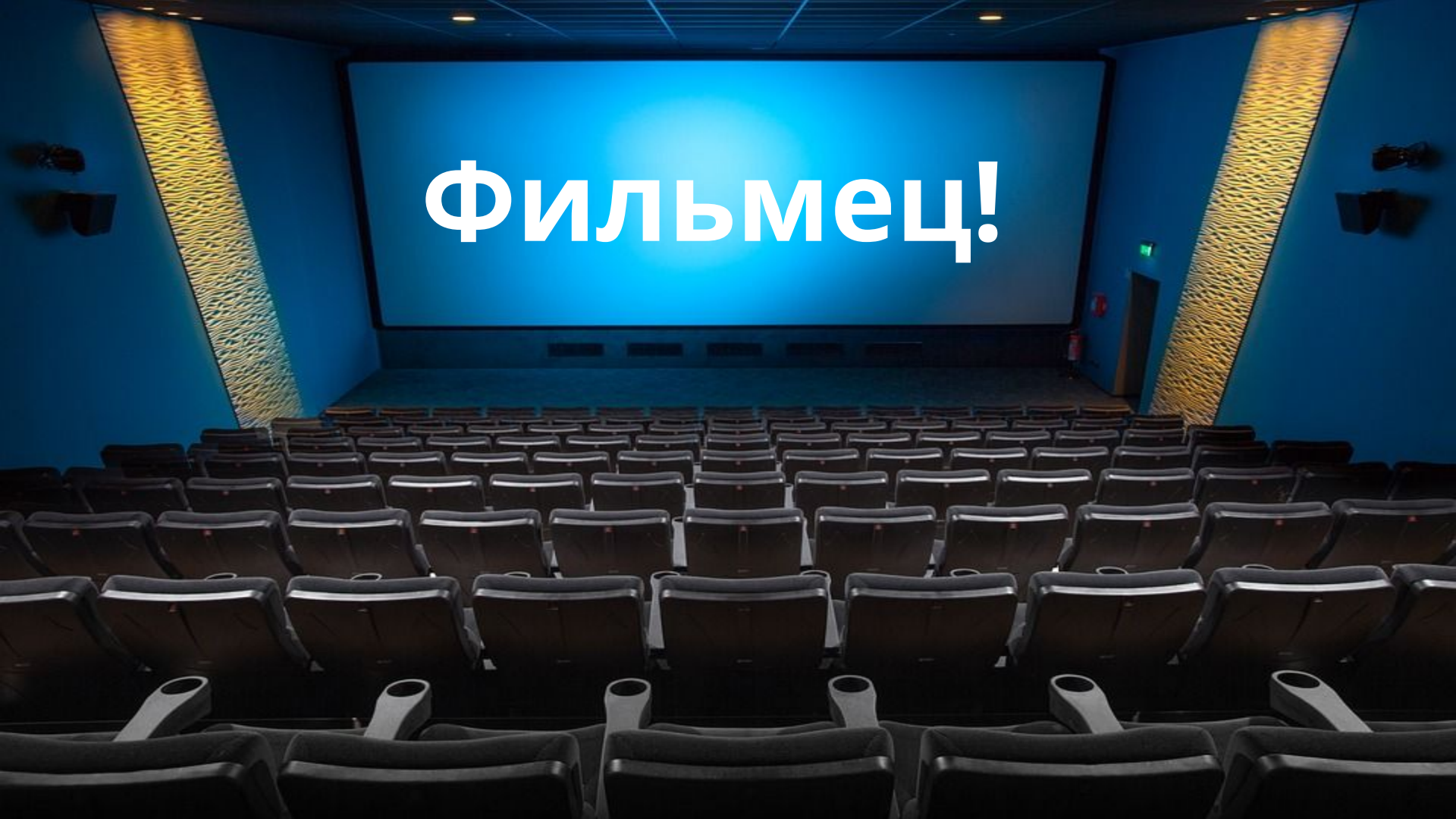
ЗаSTUBим бэкенд

Шакшина Мария

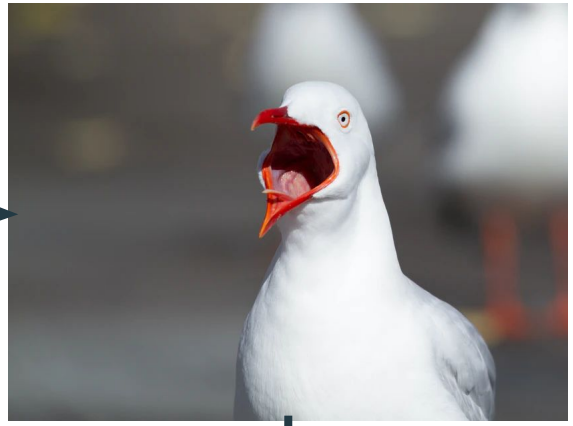
Roox Solutions

Кто я?

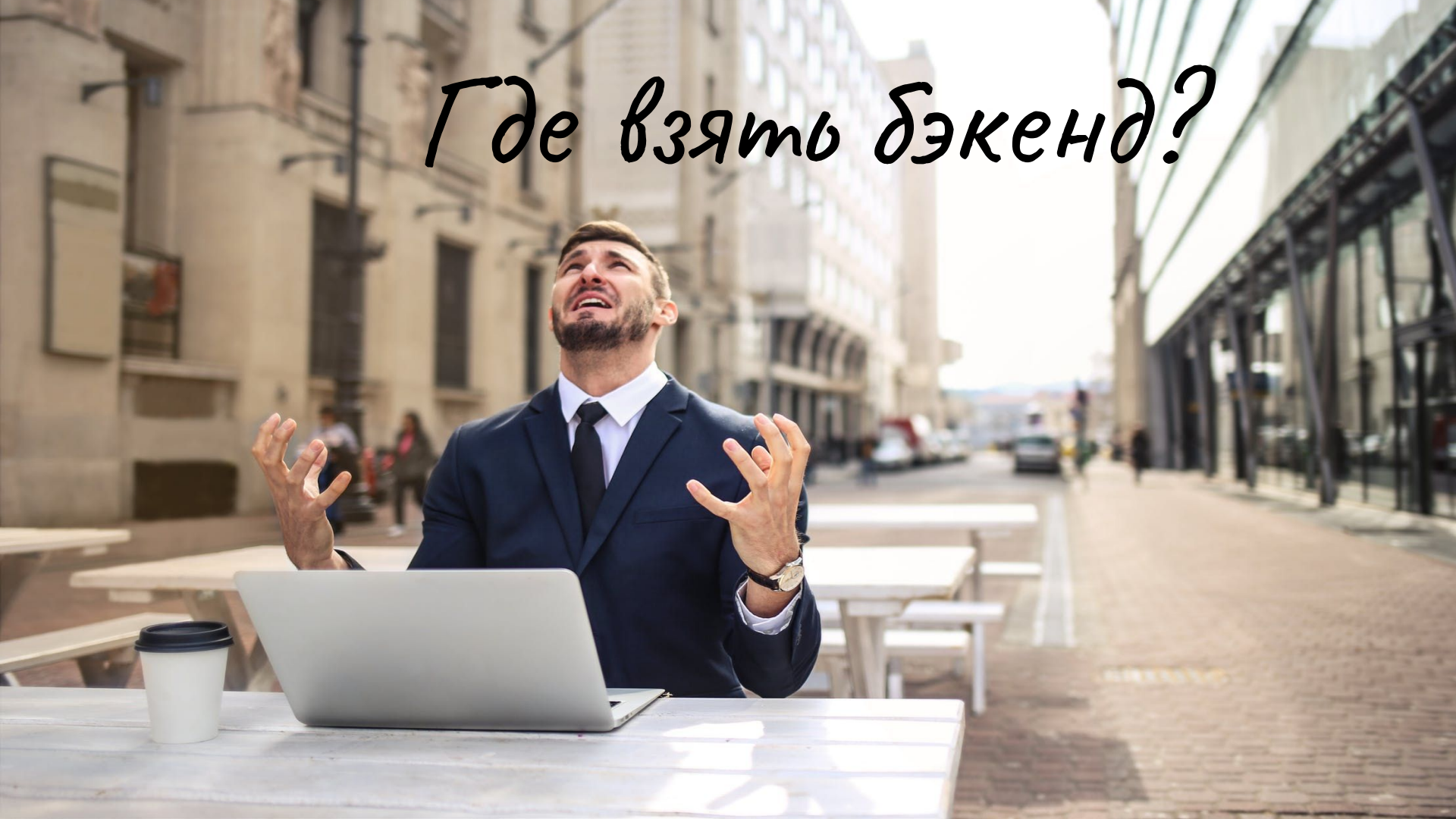
- Senior JS developer в Roox Solutions
- Пишу на React
- Опыт в разработке ~6 лет
- Интересуюсь различными направлениями в IT области


Фильмец!


Deadline




JS

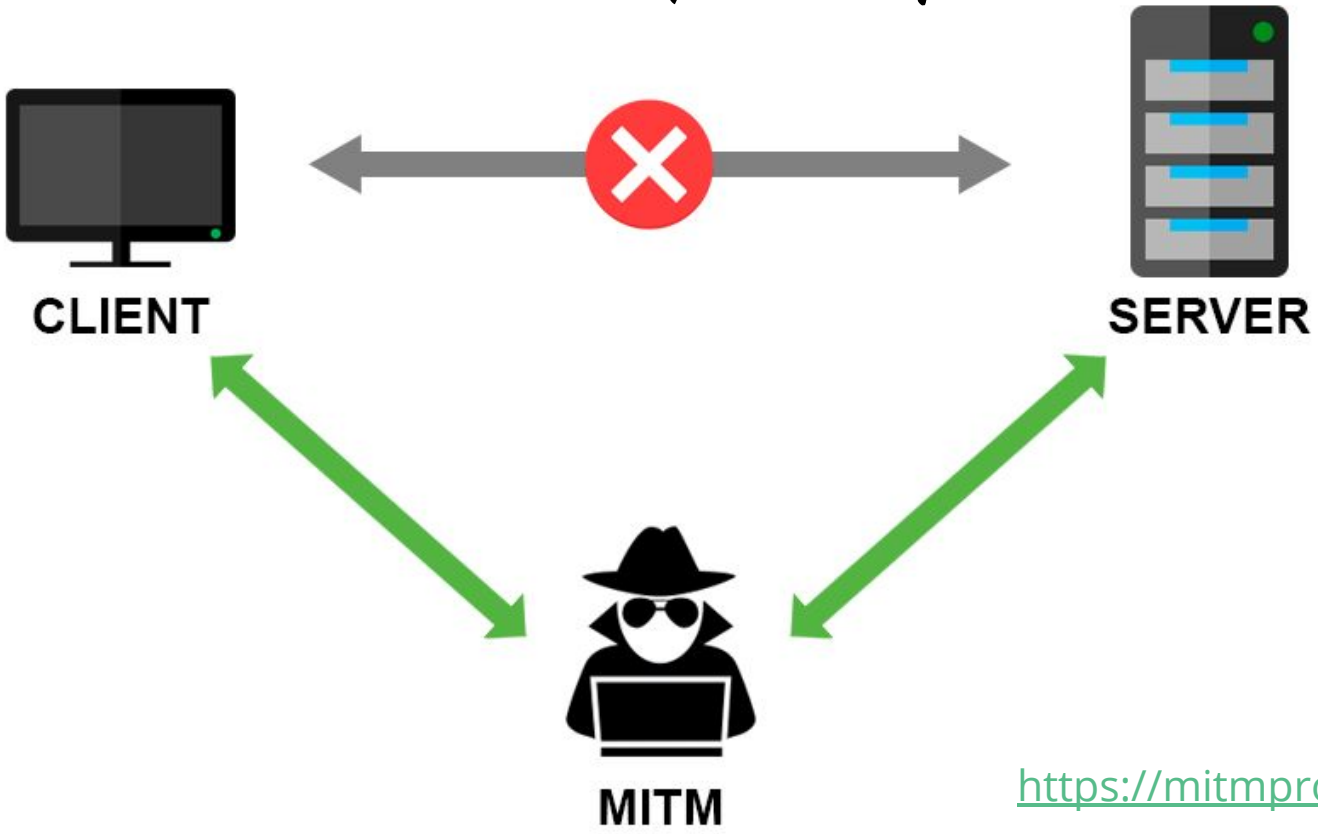

Где взять бэкенд?

Проксирование

Инструменты для проксирования

- Mitmproxy
- Fiddler
- Charles
- Server-replay

Mitmproxy



https://mitmproxy.org/


Mitmproxy

Using mitmproxy as a mock server

```python
from mitmproxy import http
import json, re

with open('user.json') as json_file:
    userData = json.load(json_file)

def request(flow: http.HTTPFlow):
    if flow.request.pretty_host == "api.github.com" and re.match('/users', flow.request.path):
        flow.response = http.HTTPResponse.make(
            200,  # (optional) status code
            json.dumps(userData),  # (optional) content
            {
                "Content-Type": "application/json",
                "Access-Control-Allow-Origin": "*"
            }  # (optional) headers
        )
```

Примеры на Python

https://github.com/mitmproxy/mitmproxy/tree/master/examples


Потрясающе!

Charles

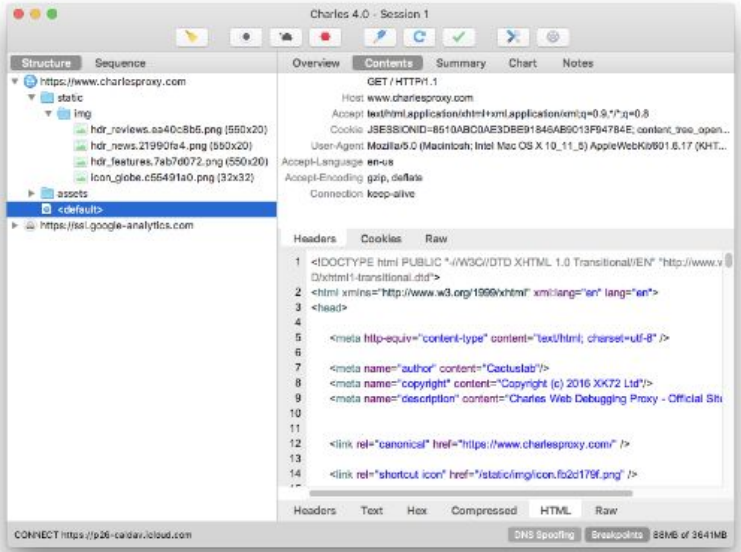
WEB DEBUGGING PROXY APPLICATION
for Windows, Mac OS and Linux

HOME | OVERVIEW | DOCUMENTATION | DOWNLOAD | BUY | SUPPORT

https://www.charlesproxy.com/

Charles is an HTTP proxy / HTTP monitor / Reverse Proxy that enables a developer to view all of the HTTP and SSL / HTTPS traffic between their machine and the Internet. This includes requests, responses and the HTTP headers (which contain the cookies and caching information).

READ MORE



DOWNLOAD

Download a free trial
Version 4.5.6

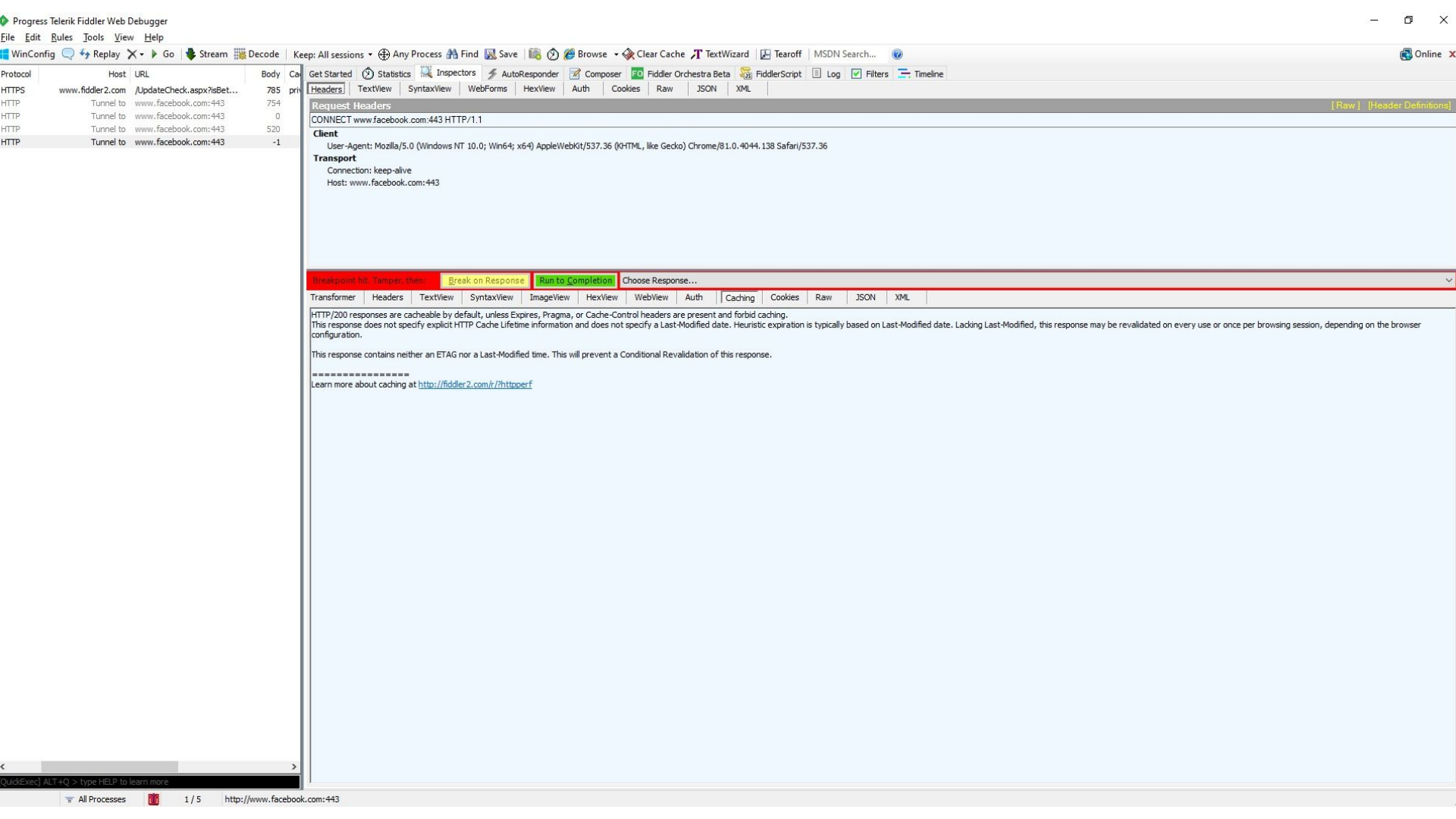
FEATURES

- Records all traffic between your browser and the Internet
- Reveals the contents of all requests, responses, cookies and headers


Charles

Fiddler

https://www.telerik.com/fiddler


Progress Telerik Fiddler Web Debugger
File Edit Rules Tools View Help
WinConfig Replay Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff MSDN Search... Online
Protocol Host URL Body
HTTPS www.fiddler2.com /UpdateCheck.aspx?isBet... 785
HTTP Tunnel to www.facebook.com:443 754
HTTP Tunnel to www.facebook.com:443 0
HTTP Tunnel to www.facebook.com:443 520
HTTP Tunnel to www.facebook.com:443 -1
Get Started Statistics Inspectors AutoResponder Composer Fiddler Orchestra Beta FiddlerScript Log Filters Timeline
Headers TextView SyntaxView WebForms HexView Auth Cookies Raw JSON XML
Request Headers [Raw] [Header Definitions]
CONNECT www.facebook.com:443 HTTP/1.1
Client
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36
Transport
Connection: keep-alive
Host: www.facebook.com:443
Breakpoint hit. Tamper, then: Break on Response Run to Completion Choose Response...
Transformer Headers TextView SyntaxView ImageView HexView WebView Auth Caching Cookies Raw JSON XML
HTTP/200 responses are cacheable by default, unless Expires, Pragma, or Cache-Control headers are present and forbid caching.
This response does not specify explicit HTTP Cache Lifetime information and does not specify a Last-Modified date. Heuristic expiration is typically based on Last-Modified date. Lacking Last-Modified, this response may be revalidated on every use or once per browsing session, depending on the browser configuration.
This response contains neither an ETAG nor a Last-Modified time. This will prevent a Conditional Revalidation of this response.
==========
Learn more about caching at http://fiddler2.com/r/?httpperf
QuickExec] ALT+Q > type HELP to learn more
All Processes 1 / 5 http://www.facebook.com:443

Progress Telerik Fiddler Web Debugger

File Edit Rules Tools View Help

#	Result	Protocol	Host	URL
659	200	HTTP	Tunnel to	da189i1jfloii.cloudfron
660	200	HTTP	Tunnel to	fonts.gstatic.com:443
661	200	HTTP	Tunnel to	logx.optimizely.com:4
662	200	HTTP	Tunnel to	beacons.gcp.gvt2.cor
663	200	HTTP	Tunnel to	www.google.ru:443
664	502	HTTP	127.0.0.1:63342	/browserConnection/t
665	200	HTTP	Tunnel to	cdn.cookielaw.org:44
666	200	HTTP	Tunnel to	dec.azureedge.net:4
667	200	HTTP	Tunnel to	script.hotjar.com:443
668	200	HTTP	Tunnel to	docs.telerik.com:443
669	200	HTTP	Tunnel to	api.dec.sitefinity.com:
670	200	HTTP	Tunnel to	stats.g.doubleclick.ne
671	502	HTTP	127.0.0.1:63342	/browserConnection/t
672	200	HTTP	Tunnel to	ssl.gstatic.com:443
673	502	HTTP	127.0.0.1:63342	/browserConnection/t
674	200	HTTP	Tunnel to	google.com:443
675	502	HTTP	127.0.0.1:63342	/browserConnection/t
676	200	HTTP	Tunnel to	play.google.com:443
677	200	HTTP	Tunnel to	play.google.com:443
678	200	HTTP	Tunnel to	play.google.com:443
679	200	HTTP	Tunnel to	beacons.gcp.gvt2.cor
680	200	HTTP	Tunnel to	play.google.com:443
681	200	HTTP	Tunnel to	beacons.gcp.gvt2.cor
682	502	HTTP	127.0.0.1:63342	/browserConnection/t
683	200	HTTP	Tunnel to	consumer.entitlement
684	502	HTTP	127.0.0.1:63342	/browserConnection/t
685	502	HTTP	127.0.0.1:63342	/browserConnection/t
686	200	HTTP	Tunnel to	browser.pipe.aria.mic
687	502	HTTP	127.0.0.1:63342	/browserConnection/t
688	502	HTTP	127.0.0.1:63342	/browserConnection/t
689	502	HTTP	127.0.0.1:63342	/browserConnection/t
690	502	HTTP	127.0.0.1:63342	/browserConnection/t
691	502	HTTP	127.0.0.1:63342	/browserConnection/t
692	502	HTTP	127.0.0.1:63342	/browserConnection/t
693	502	HTTP	127.0.0.1:63342	/browserConnection/t
694	502	HTTP	127.0.0.1:63342	/browserConnection/t
695	502	HTTP	127.0.0.1:63342	/browserConnection/t
696	200	HTTP	Tunnel to	activity.windows.com:
697	502	HTTP	127.0.0.1:63342	/browserConnection/t
698	502	HTTP	127.0.0.1:63342	/browserConnection/t
699	502	HTTP	127.0.0.1:63342	/browserConnection/t
700	502	HTTP	127.0.0.1:63342	/browserConnection/t
701	502	HTTP	127.0.0.1:63342	/browserConnection/t
702	200	HTTP	Tunnel to	d27xxe7juh1us6.clou
703	200	HTTP	Tunnel to	export.yandex.ru:443
704	200	HTTP	Tunnel to	api.lingualeo.com:443
705	502	HTTP	127.0.0.1:63342	/browserConnection/t
706	502	HTTP	127.0.0.1:63342	/browserConnection/t
707	502	HTTP	127.0.0.1:63342	/browserConnection/t
708	502	HTTP	127.0.0.1:63342	/browserConnection/t
709	502	HTTP	127.0.0.1:63342	/browserConnection/t
710	502	HTTP	127.0.0.1:63342	/browserConnection/t
711	502	HTTP	127.0.0.1:63342	/browserConnection/t

[QuickExec] ALT+Q > type HELP to learn more

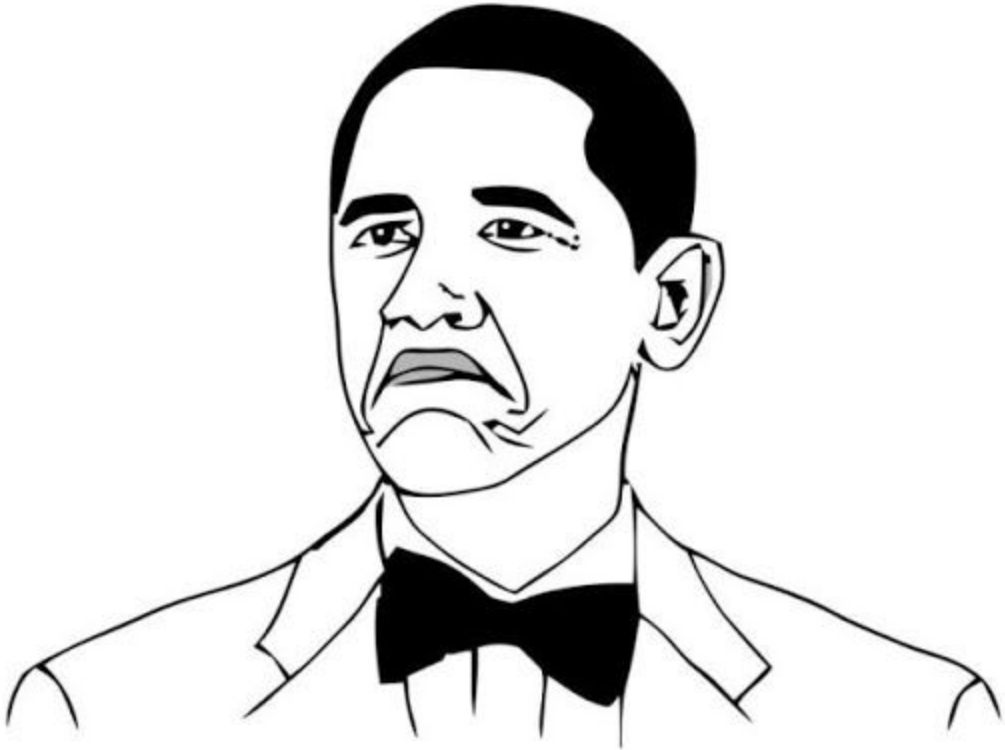
Get Started | Statistics | Inspectors | AutoResponder | Composer | Fiddler Orchestra Beta | FiddlerScript | Log | Filters | Timeline

Save Script | Go to: | Find...

```
BindPref("fiddlerscript.ephemeral.bpMethod")
public static var bpMethod: String = null;

static var bpStatus:int = -1;
static var uiBoldURI: String = null;
static var gs_ReplaceToken: String = null;
static var gs_ReplaceTokenWith: String = null;
static var gs_OverridenHost: String = null;
static var gs_OverrideHostWith: String = null;

// The OnExecAction function is called by either the QuickExec box in the Fiddler window,
// or by the ExecAction.exe command line utility.
static function OnExecAction(sParams: String[]): Boolean {

    FiddlerObject.StatusText = "ExecAction: " + sParams[0];

    var sAction = sParams[0].toLowerCase();
    switch (sAction) {
    case "bold":
        if (sParams.Length<2) {uiBoldURI=null; FiddlerObject.StatusText="Bolding cleared"; return false;}
        uiBoldURI = sParams[1]; FiddlerObject.StatusText="Bolding requests for " + uiBoldURI;
        return true;
    case "bp":
        FiddlerObject.alert("bpu = breakpoint request for uri\nbpm = breakpoint request method\nbps=breakpoint response status\nbpafter = breakpoint response for URI");
        return true;
    case "bps":
        if (sParams.Length<2) {bpStatus=-1; FiddlerObject.StatusText="Response Status breakpoint cleared"; return false;}
        bpStatus = parseInt(sParams[1]); FiddlerObject.StatusText="Response status breakpoint for " + sParams[1];
        return true;
    case "bpv":
    case "bpm":
        if (sParams.Length<2) {bpMethod=null; FiddlerObject.StatusText="Request Method breakpoint cleared"; return false;}
        bpMethod = sParams[1].toUpperCase(); FiddlerObject.StatusText="Request Method breakpoint for " + bpMethod;
        return true;
    case "bpu":
        if (sParams.Length<2) {bpRequestURI=null; FiddlerObject.StatusText="RequestURI breakpoint cleared"; return false;}
        bpRequestURI = sParams[1];
        FiddlerObject.StatusText="RequestURI breakpoint for "+sParams[1];
        return true;
    case "bpa":
    case "bpafter":
        if (sParams.Length<2) {bpResponseURI=null; FiddlerObject.StatusText="ResponseURI breakpoint cleared"; return false;}
        bpResponseURI = sParams[1];
        FiddlerObject.StatusText="ResponseURI breakpoint for "+sParams[1];
        return true;
    case "overridehost":
        if (sParams.Length<3) {gs_OverridenHost=null; FiddlerObject.StatusText="Host Override cleared"; return false;}
        gs_OverridenHost = sParams[1].toLowerCase();
        gs_OverrideHostWith = sParams[2];
        FiddlerObject.StatusText="Connecting to [" + gs_OverrideHostWith + "] for requests to [" + gs_OverridenHost + "]";
        return true;
    case "urlreplace":
        if (sParams.Length<3) {gs_ReplaceToken=null; FiddlerObject.StatusText="URL Replacement cleared"; return false;}
        gs_ReplaceToken = sParams[1];
        gs_ReplaceTokenWith = sParams[2].Replace(" ", "%20");  // Simple helper
        FiddlerObject.StatusText="Replacing [" + gs_ReplaceToken + "] in URIs with [" + gs_ReplaceTokenWith + "]";
        return true;
    case "allbut":
    case "keeponly":
        if (sParams.Length<2) { FiddlerObject.StatusText="Please specify Content-Type to retain during wipe."; return false;}
        UI.actSelectSessionsWithResponseHeaderValue("Content-Type", sParams[1]);
        UI.actRemoveUnselectedSessions();
        UI.lvSessions.SelectedItems.Clear();
        FiddlerObject.StatusText="Removed all but Content-Type: " + sParams[1];
        return true;
    case "stop":
        UI.actDetachProxy();
        return true;
    case "start":
        UI.actAttachProxy();
        return true;
    case "cls":
    case "clear":
        UI.actRemoveAllSessions();
        return true;
```

https://docs.telerik.com/fiddler/KnowledgeBase/FiddlerScript/ModifyRequestOrResponse

Capturing | All Processes | 1 / 697 | This is a CONNECT tunnel to 'www.google.com:443'. Raw Bytes Out: 1 353; In: 2 445



NOT BAD

server-replay https://www.npmjs.com/package/server-replay

Replay server responses from a .har file.

Useful if…

- …you want to develop offline and your development server isn't local
- …your development server is very slow and you want to go faster
- …you are developing against an API with rate limits

It works by starting a proxy server and serving content from a previously saved .har file overlayed with files from your local system, configurable with mappings.

Installation

```
npm install -g server-replay
```

Running

You need to have a `.har` file, run server-replay, and then set up your browser to use it as a proxy.

A `.server-replay.json` in the current directory is used by default if no config option is given.

```
server-replay [options] <.har file>

Options:
  -c, --config  The config file to use
  -p, --port    The port to run the proxy server on  [default: 8080]
  -d, --debug   Turn on debug logging
```

Install

```
> npm i server-replay
```

Weekly Downloads

6

Version
1.1.0

License
Apache-2.0

Issues
3

Pull Requests
2

Homepage
github.com/Stuk/server-replay

Repository
github.com/Stuk/server-replay

Last publish
4 years ago

Collaborators

Try on RunKit

Report a vulnerability

https -> http

```
{
    "version": 1,
    "replacements": [
        // Proxy only works over http
        {"match": "https", "replace": "http"}
    ]
}
```

http-proxy-middleware

src/setupProxy.js

```
const { createProxyMiddleware } = require('http-proxy-middleware');
module.exports = function(app) {
  app.use(
    '/api',
    createProxyMiddleware({
      target: 'http://localhost:5000',
      changeOrigin: true,
    })
  );
};
```

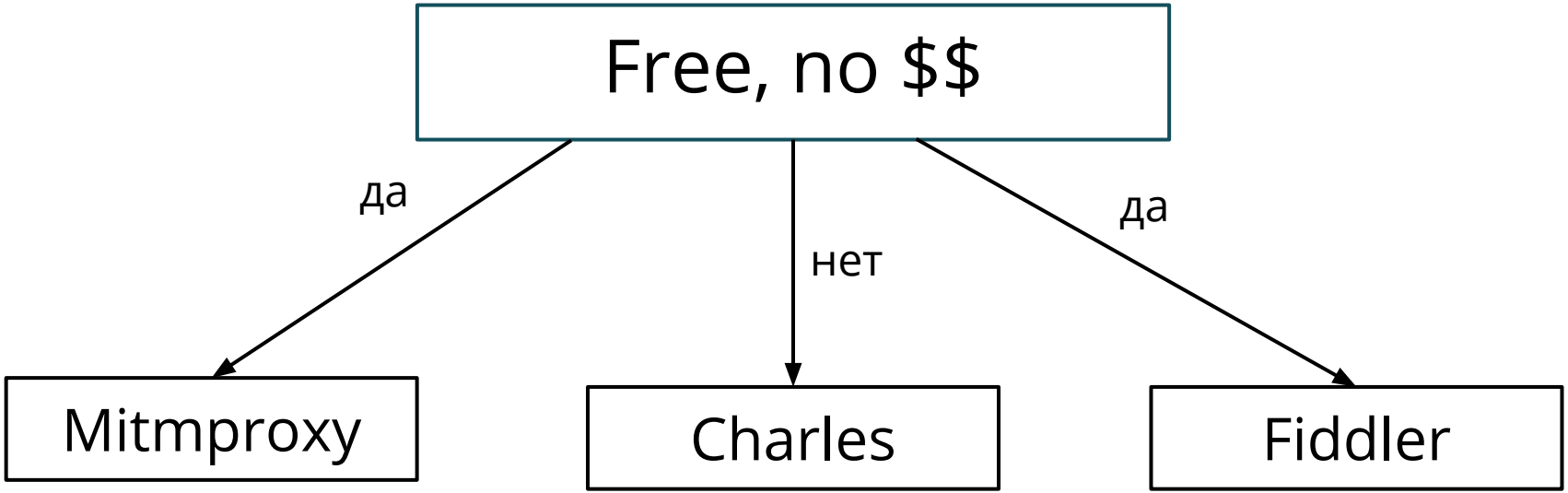
https://create-react-app.dev/docs/proxying-api-requests-in-development/

Invalid Host header и др проблемы

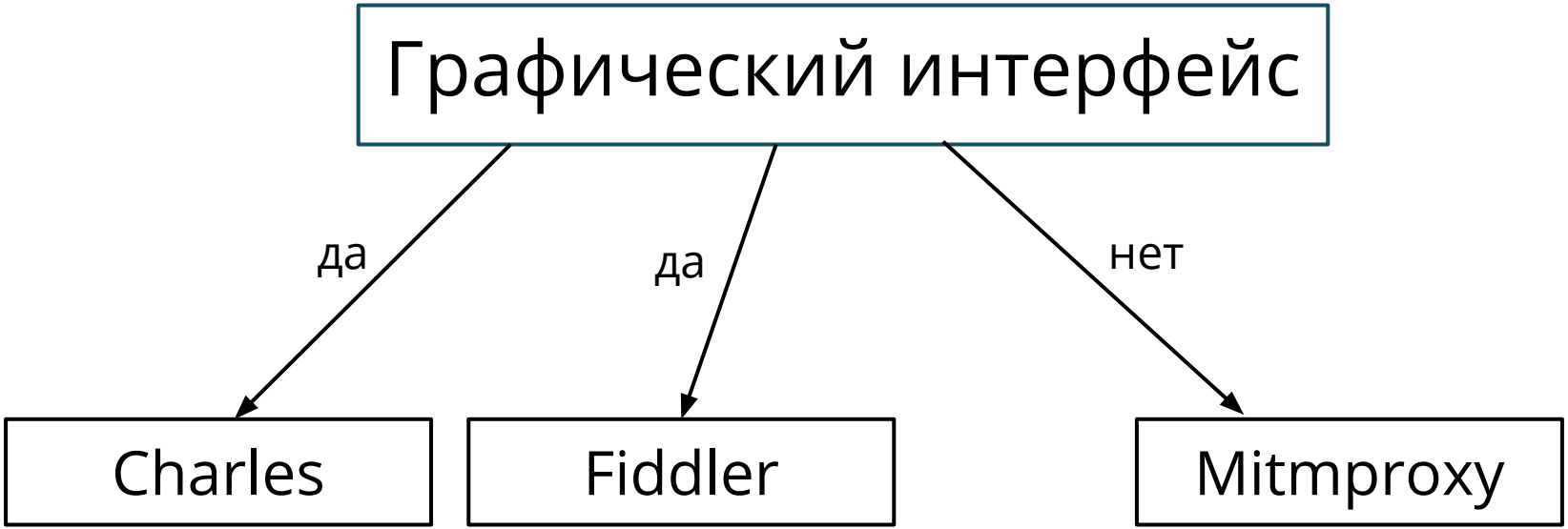
.env.development

```
HOST=local.devhost.com
```

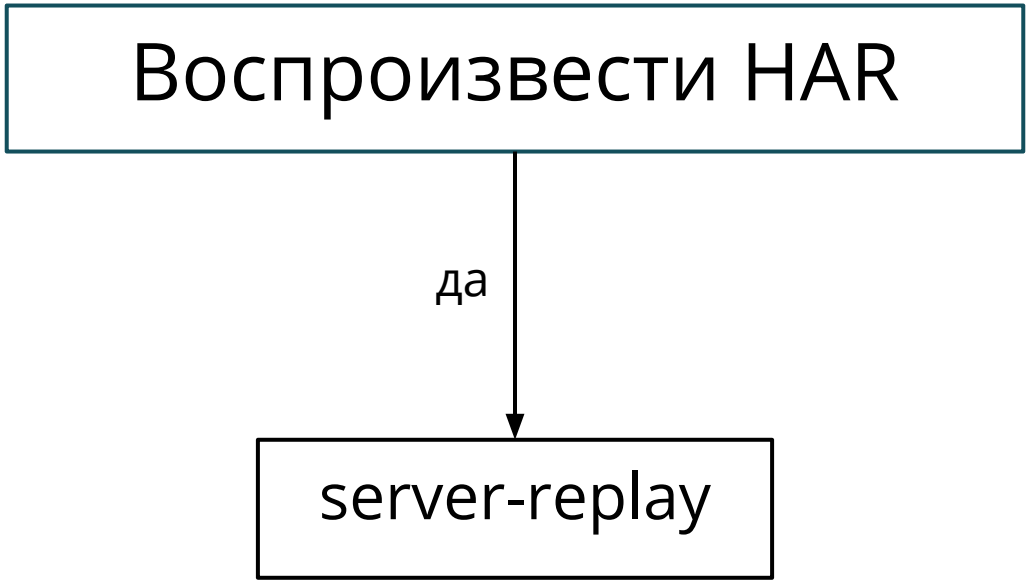
Стоимость



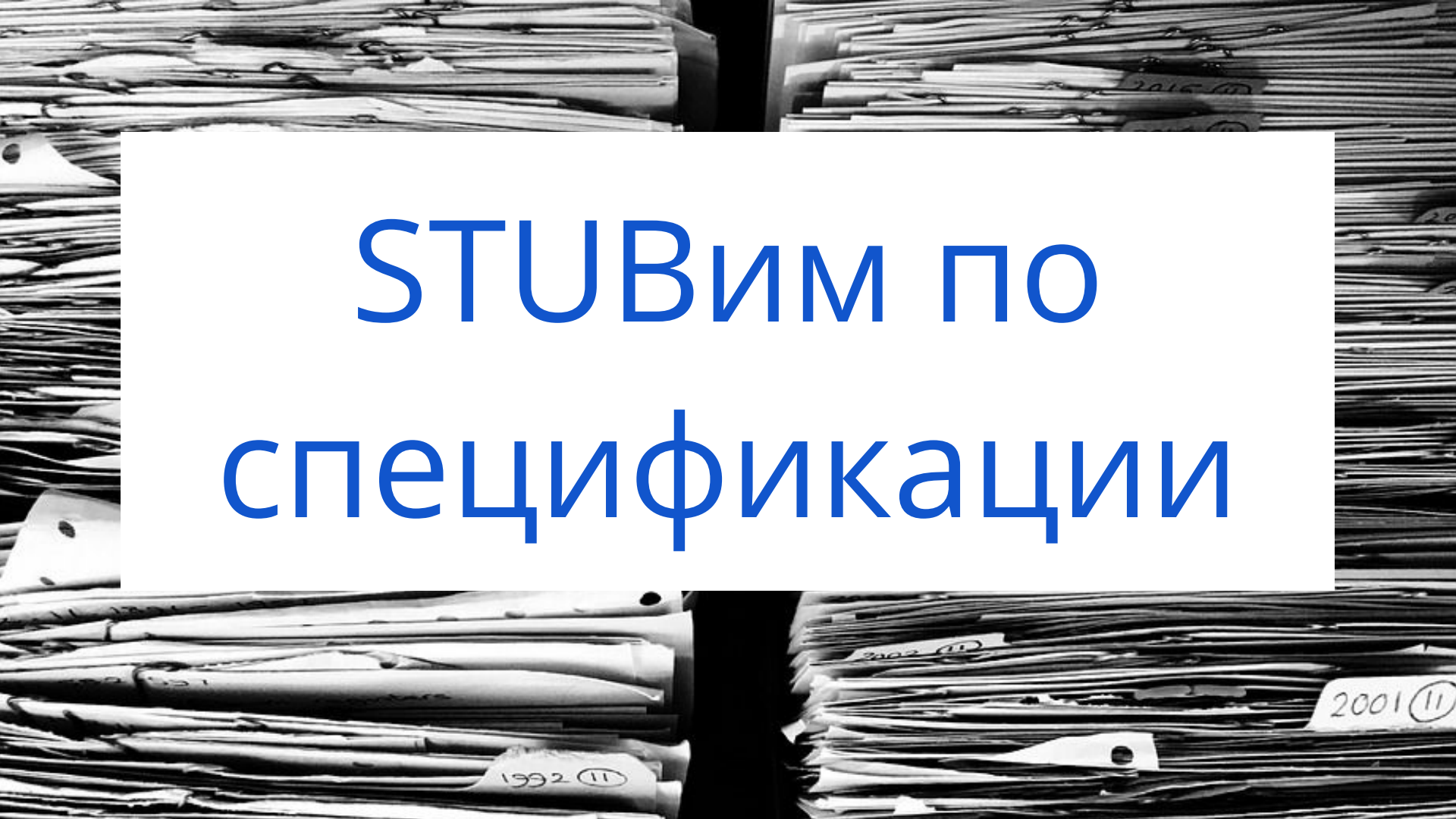
Наличие UI


Графический интерфейс
да
да
нет
Charles
Fiddler
Mitmproxy

Ответы из HAR-ника



Проксирование во фреймворке

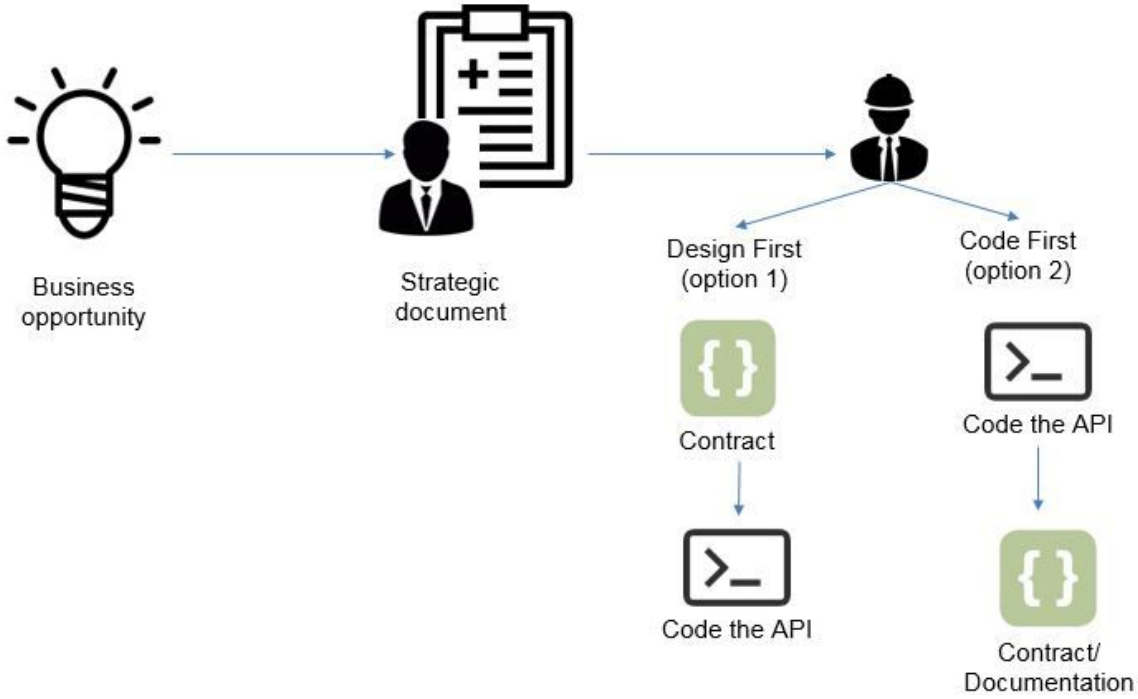


STUBим по спецификации

Виды спецификаций

- OpenAPI v2/v3;
- API Blueprint;
- RAML

Design First vs Code First



Инструменты для OpenAPI

- Swagger;
- SoapUI;
- Stoplight

Mock Servers

https://openapi.tools/

Fake servers that take description document as input, then route incoming HTTP requests to example responses or dynamically generates examples.

Name	Language	v2	v3	GitHub
API Sprout - Lightweight, blazing fast, cross-platform OpenAPI 3 mock server with validation	cli / Docker	❌	✅	
Apitive Studio - A platform for Digital Product Managers and API Consultants to design REST APIs with in-built mock and documentation.	Angular 7.0, Java / Saas	✅	✅	
Connexion - OpenAPI First framework for Python on top of Flask with automatic endpoint validation & OAuth2 support	Python	✅	✅	
Fakeit - Create mock server from OpenAPI 3 specification with random response generation and request validation.	cli / Docker	❌	✅	
Falcon Heavy - The framework for building app backends and microservices via the API design-first workflow.	Python	❌	✅	
Meeshkan - Mock HTTP APIs through a combination of API definitions, recorded traffic and code. Used for sandboxes, as well as automated and exploratory testing.	Python	❌	✅	
Microcks - Mocking and testing platform for API and microservices. Turn your OAI contract examples into ready to use mocks. Use examples to test and validate implementations according schema elements.	Self-hosted / SaaS	✅	✅	

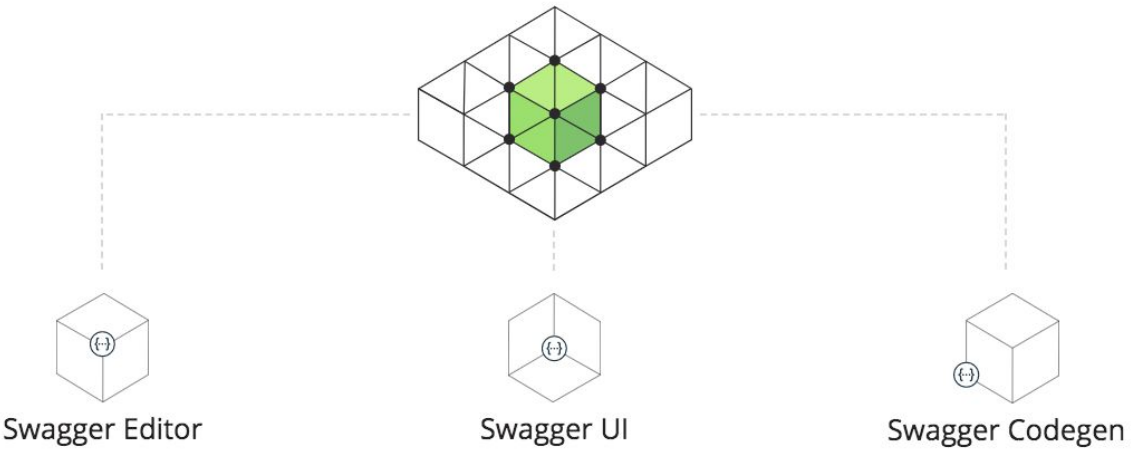
Swagger

https://swagger.io/

SwaggerHub

The design and documentation platform for teams and individuals working with the OpenAPI Specification.

Create Free Account




Swagger

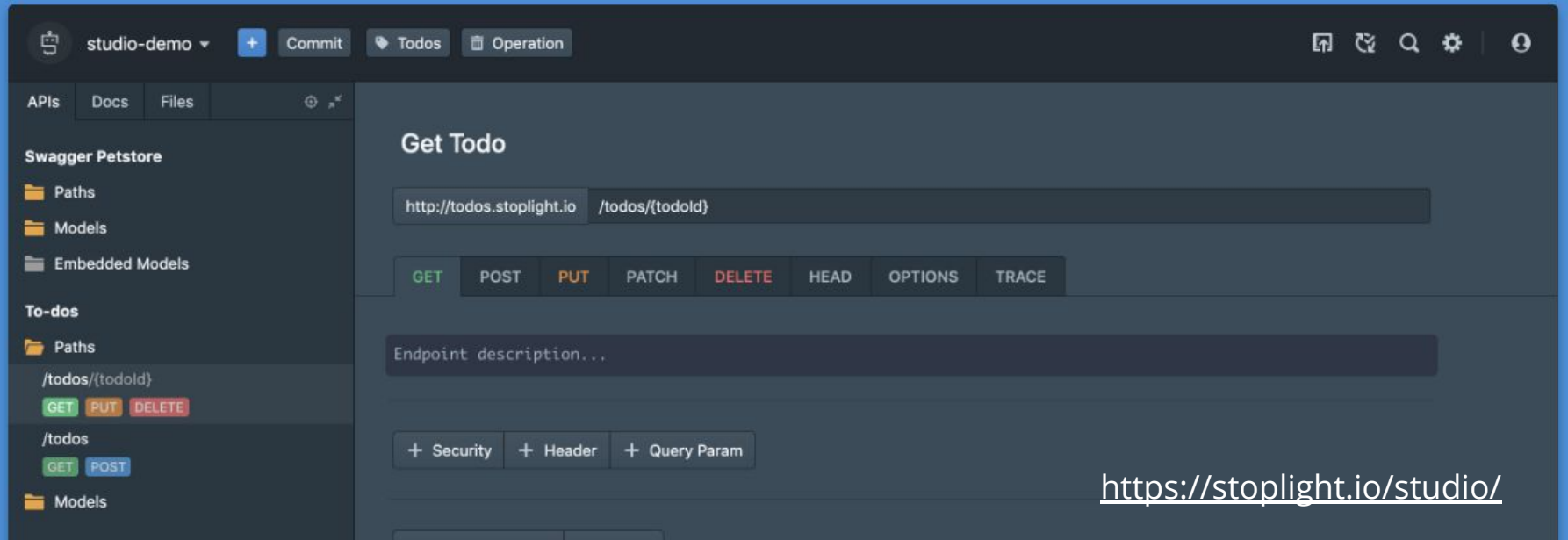
SoapUI

https://www.soapui.org/

Stoplight Studio

Design APIs 10x faster with our free OpenAPI editor. Prototype and share your API within minutes.

On the Web | Mac App | Windows App | Linux App | Read the Docs



https://stoplight.io/studio/

Prism, an Open-Source HTTP Mock & Proxy Server

Accelerate API development with realistic mock servers, powered by OpenAPI documents.

GitHub Watch the Video Read the Docs

Quick Iterations Dynamic Examples Validation Callbacks Proxy

Work with your API before you write any code

Prism is an open-source HTTP mock server that can mimic your API's behavior as if you already built it. Mock HTTP servers are generated from your OpenAPI v2/v3 (formerly known as Swagger) documents.


Stoplight &
Prism

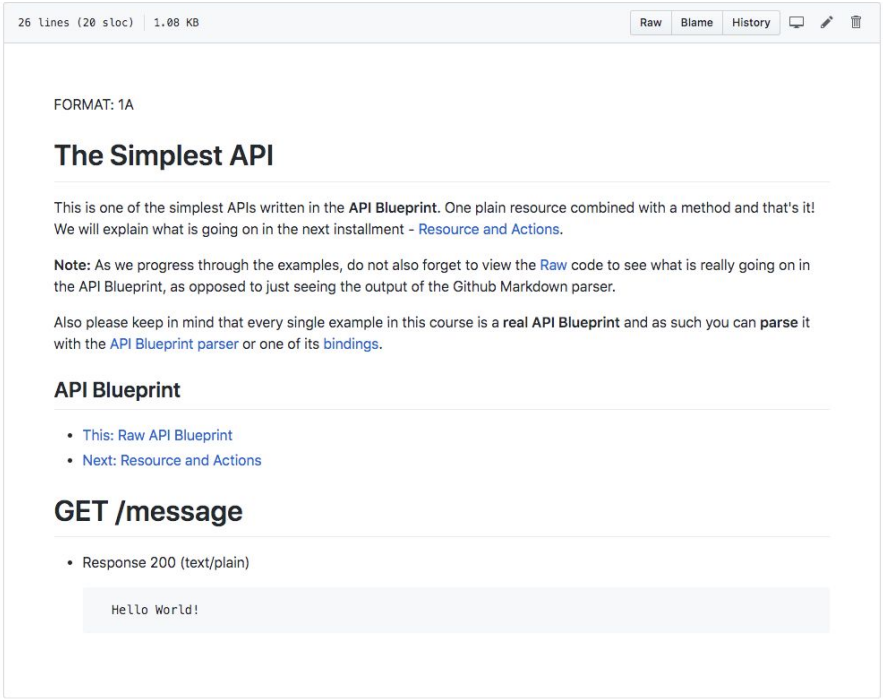
Swagger to ts

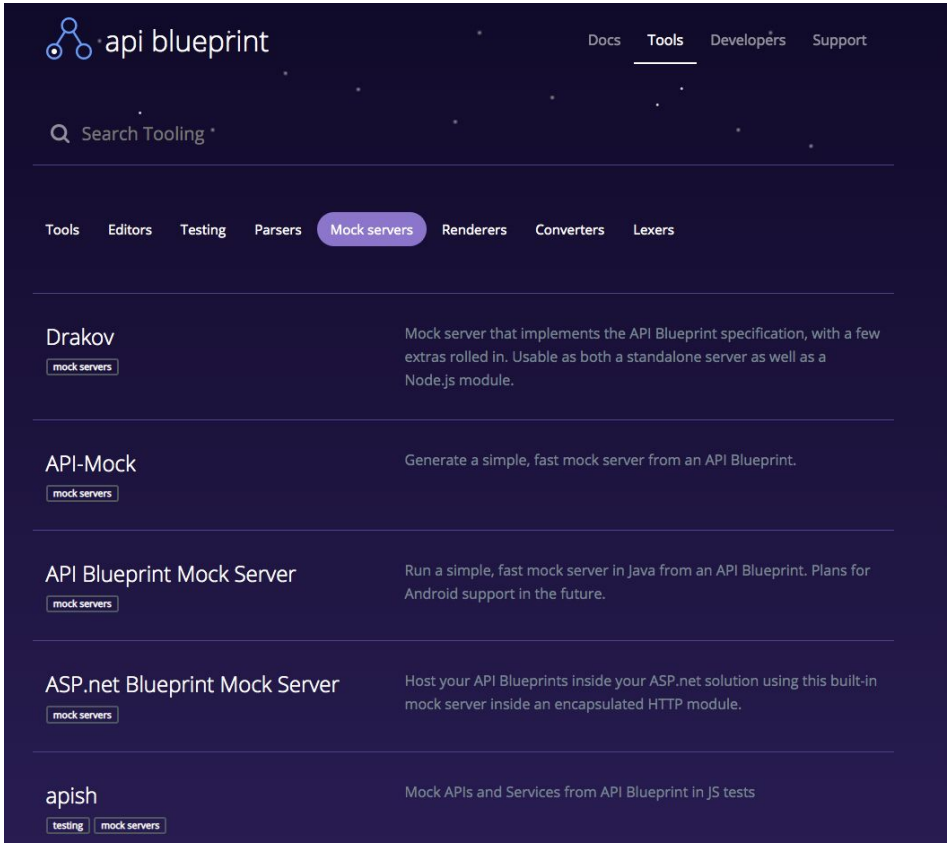
```
npx @manifoldco/swagger-to-ts schema.yaml --output schema.ts
```

https://github.com/manifoldco/swagger-to-ts

Спецификация API Blueprint

https://apiblueprint.org/tools.html





Плюсы спецификаций

Есть файл со спецификацией ==
есть мок-сервер

Минусы спецификаций

Спецификация может быть необновленной => неактуальный мок-сервер


Спецификация!

Fake your data

JS-библиотеки для фейковых данных

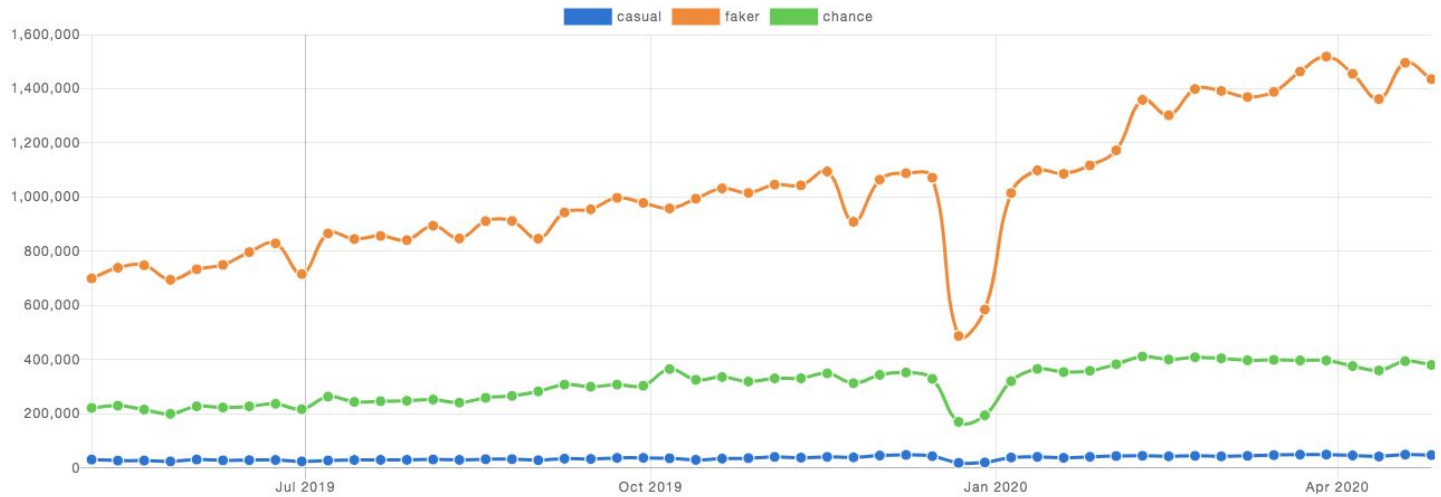
- Faker.js https://github.com/Marak/faker.js;
- Chancejs https://chancejs.com/index.html;
- Casual https://github.com/boo1ean/casual;

Пример использования Faker.js

```
users.push({
    id: faker.random.uuid(),
    fullName: faker.name.findName(),
    email: faker.internet.email(),
    country: faker.address.country(),
    phoneNumber: `+7${faker.phone.phoneNumber()}`
});
```

Сравнение fake data библиотек

Downloads in past 1 Year



Stats

	stars	forks	issues	updated	created	size
casual	2,519	143	33	Mar 23, 2020	Jan 31, 2014	minzipped size 178.2 KB
faker	23,418	1,933	321	May 6, 2020	May 15, 2010	minzipped size 321.1 KB
chance	5,115	363	120	Apr 30, 2020	Jun 21, 2013	minzipped size 67.9 KB

https://www.npmtrends.com/

Готовим STUBы

SinonJs FakeServer

```
const fakeServerWrapper = {
    init: function() {
        faker.locale = "ru"; //Устанавливаем локаль

        // Создаем сервер
        this.fs = sinon.createFakeServer({autoRespond: true});
        this.fs.xhr.useFilters = true;

        // Если фильтр возвратит true, запрос не будет сэмулирован,
        // а пойдет на настоящий сервер
        this.fs.xhr.addFilter(
            function(method, url, async, username, password) {
                return (new RegExp(externalHost)).test(url);
        });

        this.fs.respondWith("GET", url,
            (xhr) => {
                const users = generateUsers(3);
                xhr.respond(200, //Устанавливаем любой статус код ответа
                    { "Content-Type": "application/json" }, JSON.stringify(users));
            }
        );
    },

    restore: function() {
        this.fs.restore();
    }
};
```

https://sinonjs.org/releases/latest/fake-xhr-and-server/

Поддержка Fetch API

Поддерживает только **XMLHttpRequest** объекты. Но можно использовать фейковый Fetch.
https://www.npmjs.com/package/fake-fetch

FakeServer

JSON Server

https://github.com/typicode/json-server

- Создаем файл *db.json*
- Запускаем сервер
  `json-server --watch db.json`
- Дергаем, например,
  http://localhost:3000/comments/1
- Получаем в ответе:

```
{ "id": 1, "body": "Some new comment", "postId": 1 }
```

db.json

```
{
  "posts": [
    {
      "id": 1,
      "title": "json-server",
      "author": "typicode"
    }
  ],
  "comments": [
    {
      "id": 1,
      "body": "Some new comment",
      "postId": 1
    }
  ],
  "profile": {
    "name": "Mary"
  }
}
```

Запуск JSON Server

node server.js - запустить сервер.

npx nodemon server.js - запустить сервер со слежением за изменением файлов.

```
const server = jsonServer.create();
```

```
const router = jsonServer.router('data.json');
```

```
// Добавить кастомные пути в роутер (/operations?accountNumber={number})
server.get('/operations', (req, res) => {
    res.jsonp(getOperations(req.query.accountNumber))
});
```

```
// Добавить правила в роутер
server.use(jsonServer.rewriter({
    '/api/*': '/$1',
}));
```

```
server.use(router);
```

```
server.use((req, res, next) => {
    if (isAuthorized(req)) { // Добавить свою логику авторизации
        next() // Продолжаем
    } else {
        res.sendStatus(401);
    }
});
```

```
server.post('/messages', (req, res, next) => {
    const userId = req.body['senderId'];
    // Проверим, передан ли senderId
    if (userId) {
        req.body.createdAt = moment().format();
    } else {
        // Проставить свой статус код ответа
        res.status(400).jsonp({
            error: "No valid senderId"
        });
    }
    next();
});
```

My JSON Server

Fake Online REST server for teams

Create a **JSON file** on **GitHub**

github.com/user/repo/master/db.json

```
{
  "posts": [
    {
      "id": 1,
      "title": "hello"
    }
  ],
  "profile": {
    "name": "typicode"
  }
}
```

Get **instantly** a **fake server**

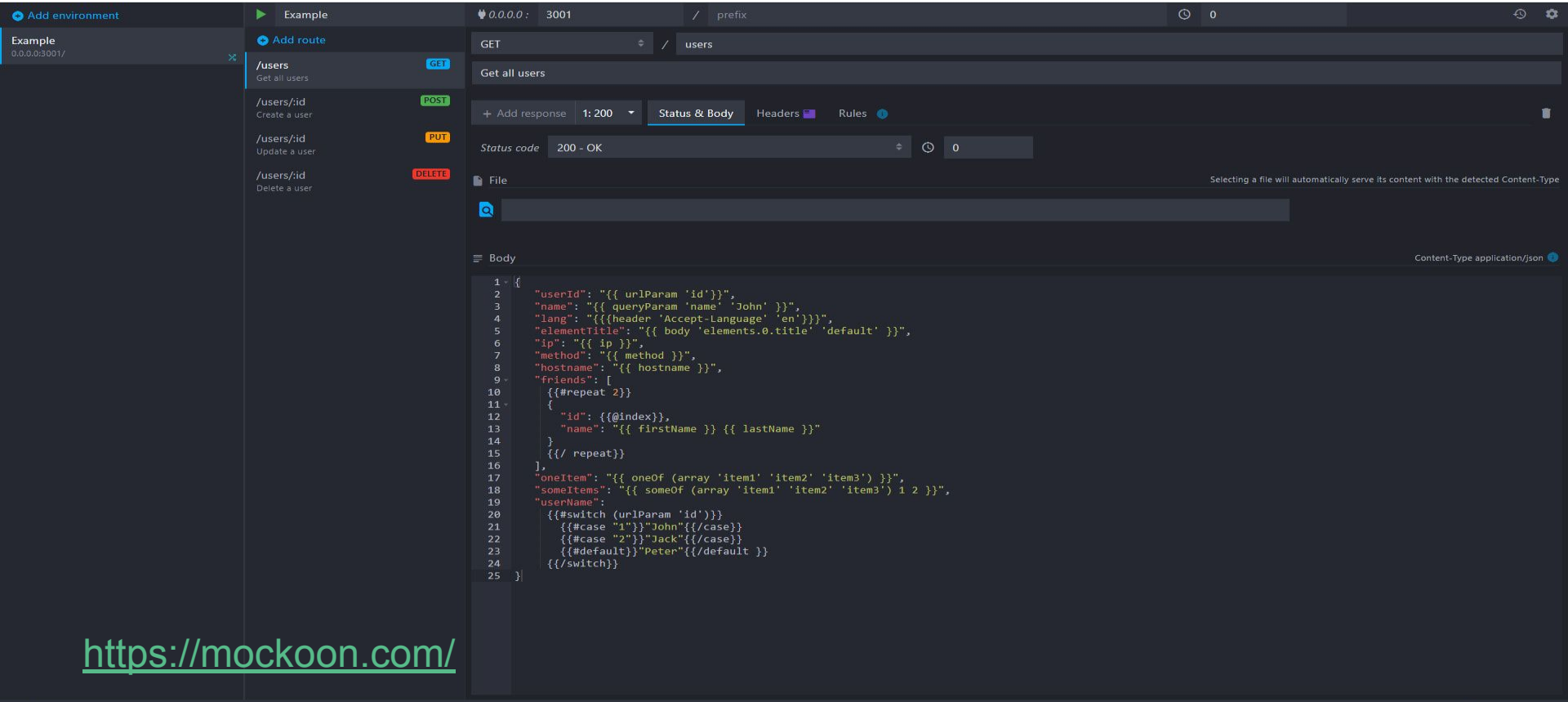
my-json-server.typicode.com/user/repo/posts/1

```
{
  "id": 1,
  "title": "hello"
}
```

https://my-json-server.typicode.com

JSON Server

mockoon



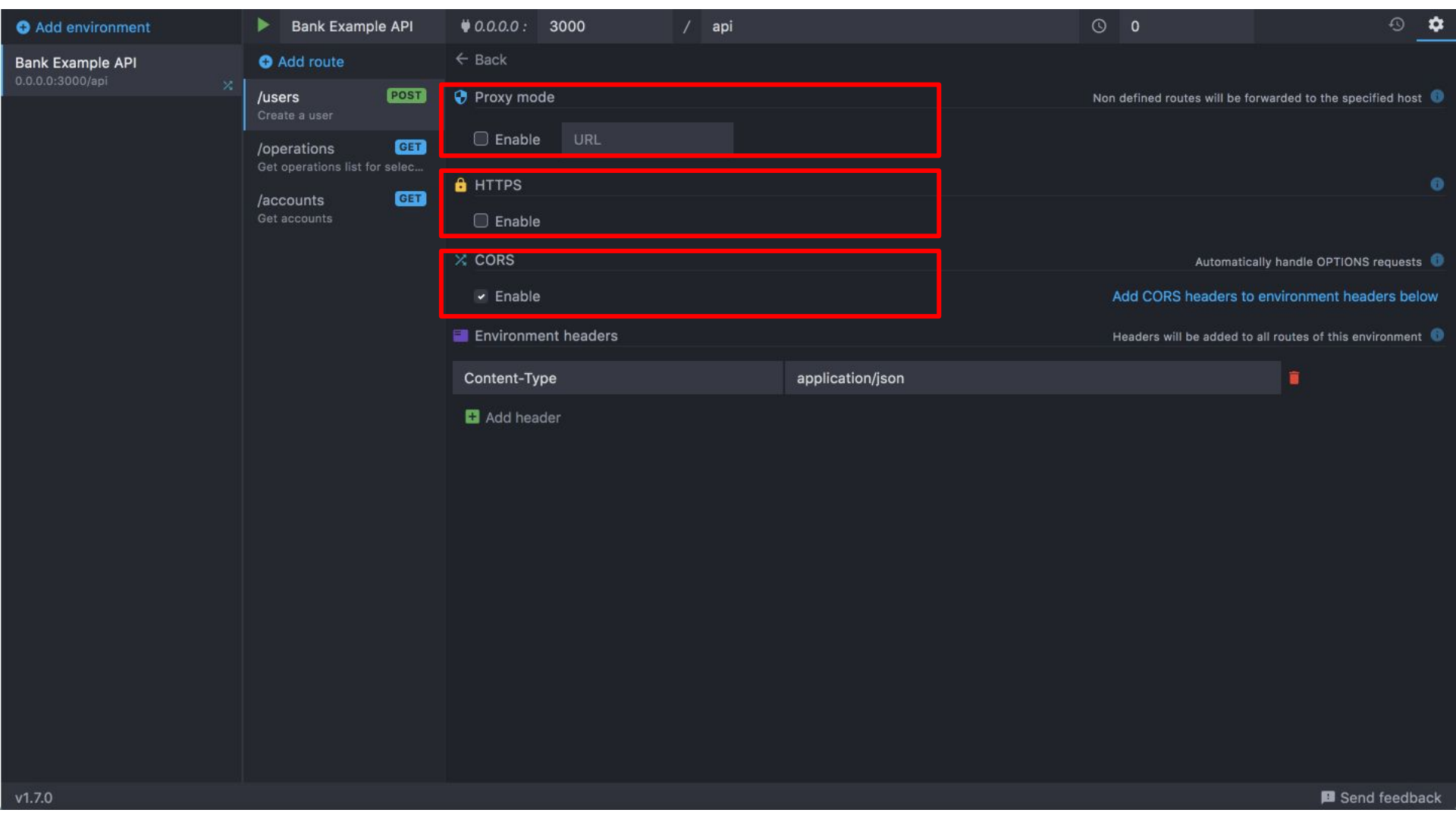
https://mockoon.com/

v1.5.0

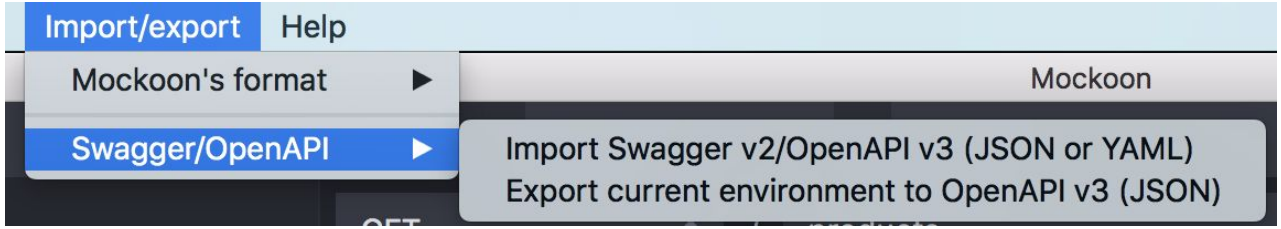
Send feedback


Mockoon

```
{{#repeat 2}}
    {
        "date": "{{date '2020-06-14' '2020-06-10' 'YYYY-MM-DD'}}",
        "currency": "810",
        "documents": [
            {{#repeat 3}}
                {
                    "documentId": "{{guid}}",
                    "documentNum": "{{int 0 1000}}",
                    "amount": "-{{float 0 100000 round=0.01}}",
                    "paymentDesc": "{{lorem}}",
                    "accountNumber": "{{queryParam 'accountNumber'}}"
                }
            {{/repeat}}
        ]
    }
{{/repeat}}
```


Add environment
Bank Example API
0.0.0.0:3000/api
Bank Example API
0.0.0.0 :
3000
/
api
0
Add route
/users
POST
Create a user
/operations
GET
Get operations list for selec...
/accounts
GET
Get accounts
Back
Proxy mode
Non defined routes will be forwarded to the specified host
Enable
URL
HTTPS
Enable
CORS
Automatically handle OPTIONS requests
Enable
Add CORS headers to environment headers below
Environment headers
Headers will be added to all routes of this environment
Content-Type
application/json
Add header
v1.7.0
Send feedback

Есть импорт, нет синхронизации



Где же баг?


mockoon

Стабить прямо на фронте

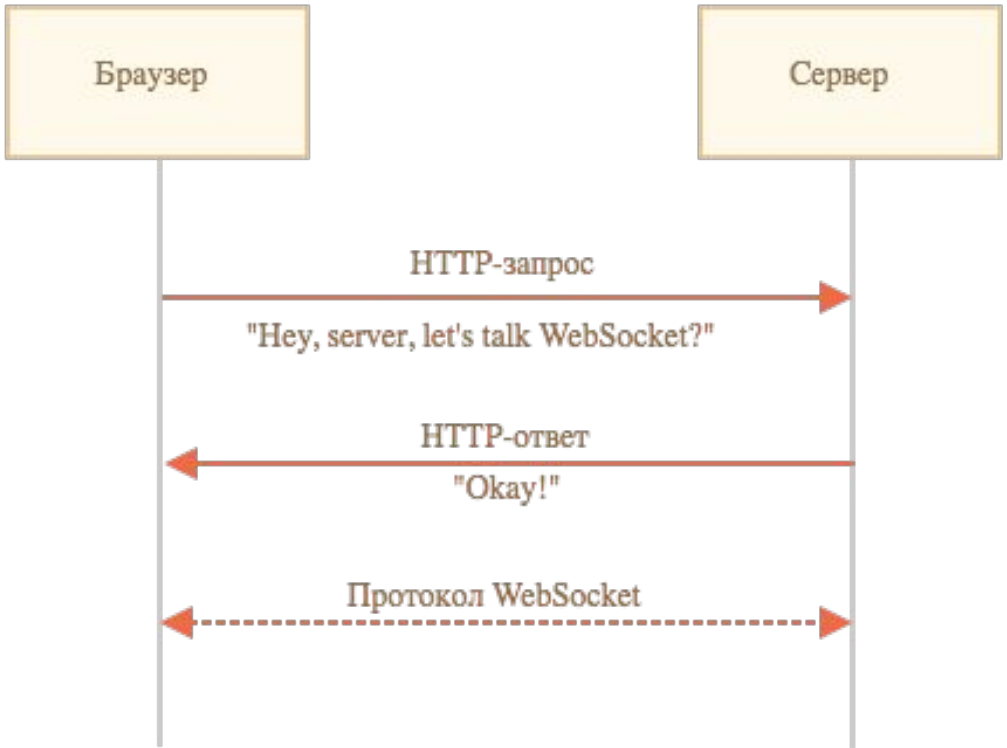
https://clck.ru/Md3oX

Стаб на фронте из практики

```
export const someMethod = (params = {}) => {
  if (useStub) {
    return fetchStub('someMethod');
  }
  …
}
```

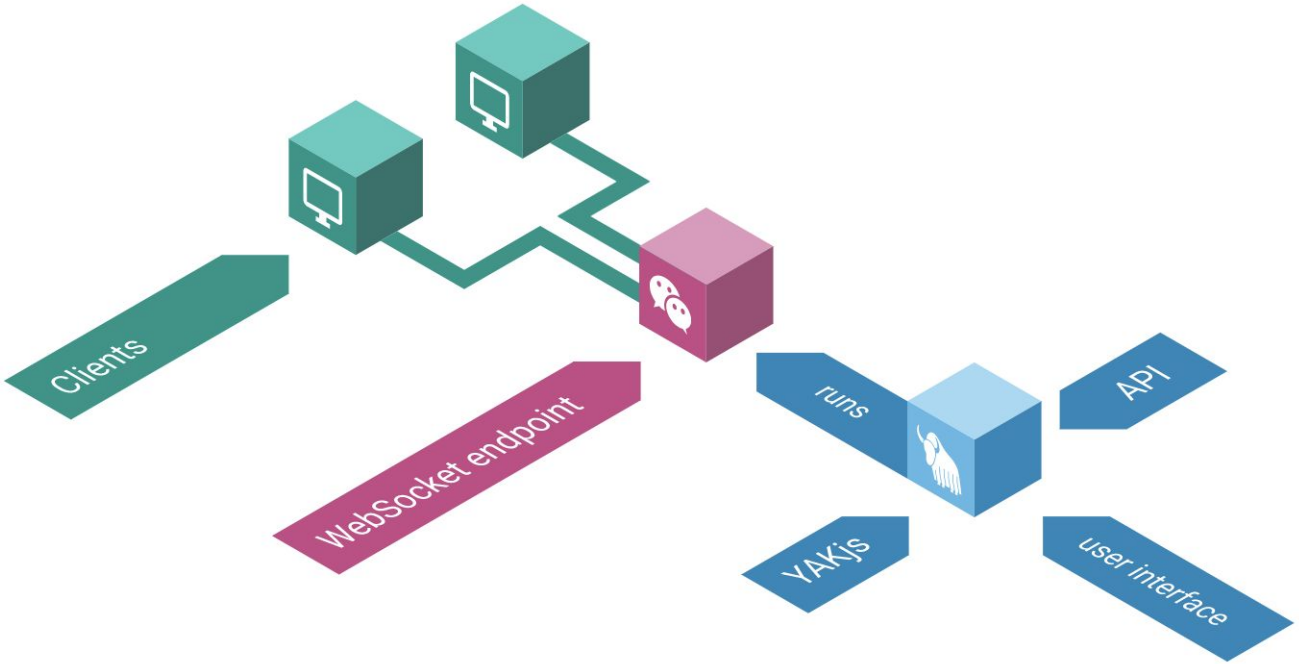
```
export function* fetchStub(methodName) {
  yield wait(1000);
  return data[methodName];
}
```

WebSocket


Браузер
Сервер
HTTP-запрос
"Hey, server, let's talk WebSocket?"
HTTP-ответ
"Okay!"
Протокол WebSocket

YAKjs

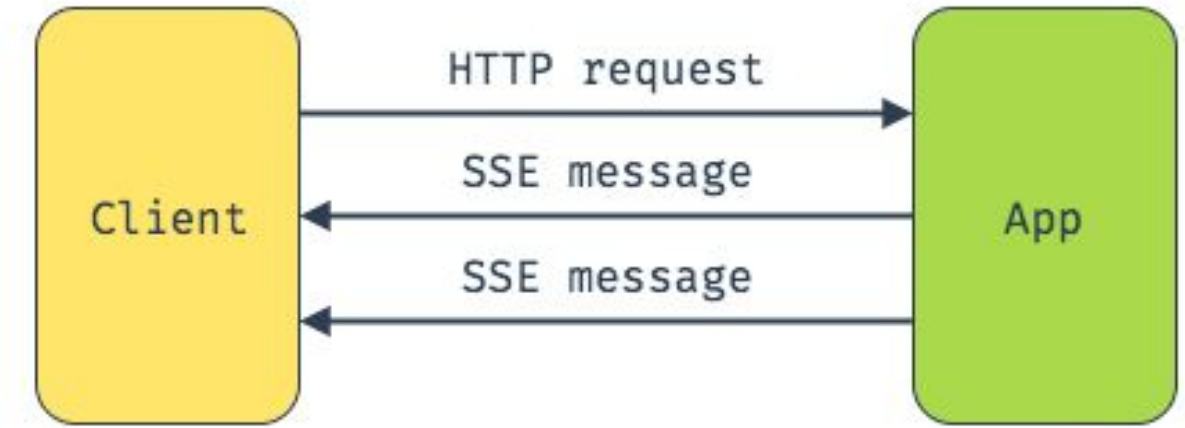
http://www.yakjs.com/




YAKjs

И здесь тоже нет синхронизации

Server-Sent Events



local-web-server

Инструмент на все случаи жизни:

- REST API,
- WebSocket,
- Server Sent Events service

SSE

```
{
    route: '/notifications',
    responses: [
        {
            response: async function (ctx) {
                ctx.body = new require('stream').PassThrough();
                ctx.type = 'text/event-stream';
                ctx.set('Cache-Control', 'no-cache');
                ctx.set('Connection', 'keep-alive');

                const interval = setInterval(() => {
                    ++i;
                    ctx.body.write(`event: message\n`);
                    ctx.body.write(`data: ${JSON.stringify(getNotification(i))}\n\n`)
                }, 3000);

                function finished () {
                    clearInterval(interval);
                    ctx.body.end();
                }

                ctx.req.on('close', finished);
                ctx.req.on('finish', finished);
                ctx.req.on('error', finished);
            }
        }
    ]
}
```

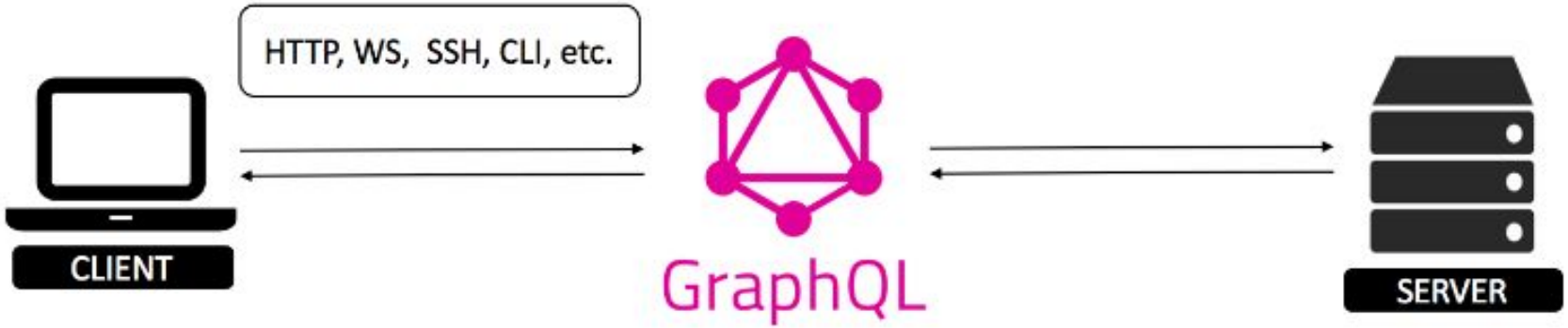
WebSocket

```
class Websocket {
    middleware (config, lws) {
        const WebSocket = require('ws');
        const wss = new WebSocket.Server({ server: lws.server });

        wss.on('connection', socket => {
            socket.on('message', message => {
                socket.send(`"${message}"? Yep, that's right`);
            })
        })
    }
}

module.exports = Websocket;
```

REST API

```
const router = require('koa-route');
return [
    router.get('/users', function (ctx) {
        if (isAuthorized()) {
            ctx.response.type = 'json';
            ctx.response.body = users;
        } else {
            ctx.response.status = 401;
        }
    }),
    router.post('/users', function (ctx) {
        const newUser = ctx.request.body;
        users.push(newUser);
        newUser.id = users.length;
        ctx.response.status = 201;
        ctx.response.set('Location', `/users/${newUser.id}`);
    }),
    router.put('/users', function (ctx) {
        ctx.response.status = 405;
    }),
    router.get('/users/:id', function (ctx, id) {
        ctx.response.type = 'json';
        ctx.response.body = users.find(user => user.id === Number(id));
    })
]
```

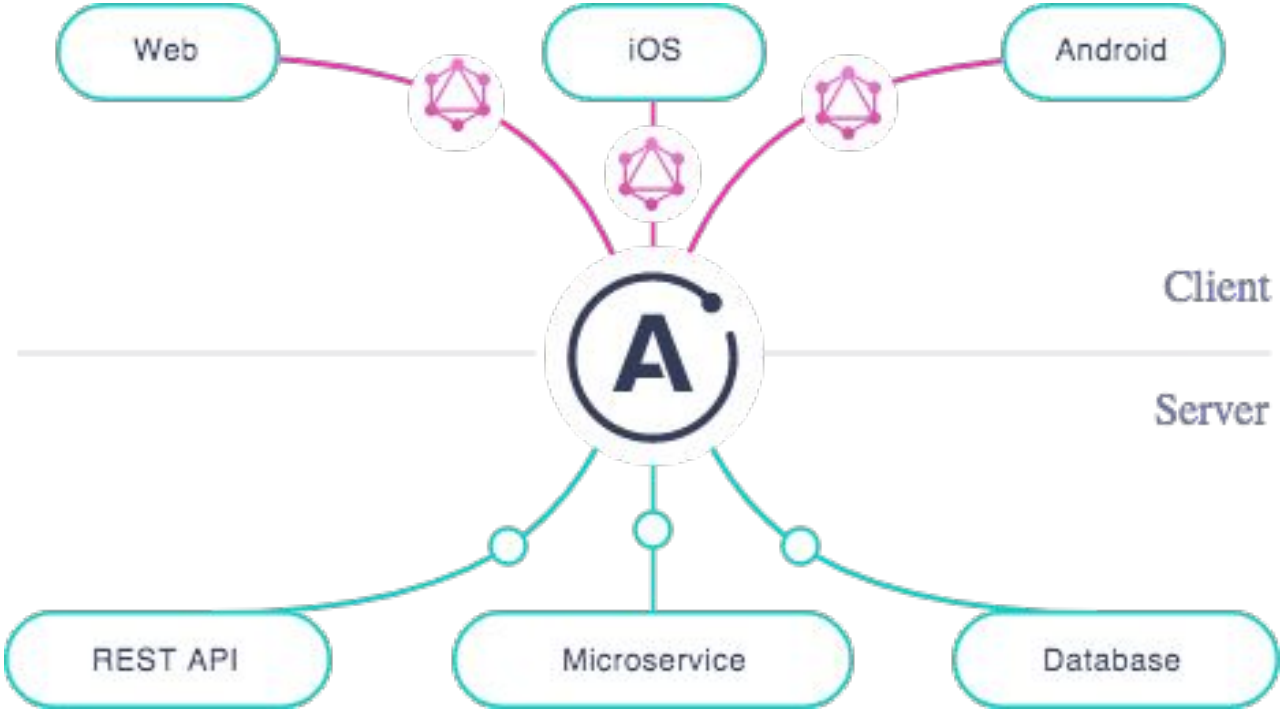
GraphQL



„GraphQL на проде похож на привидение: все о нем говорят, но мало кто видел.“

— Франсуа де Ларошфуко, «Максимы и моральные размышления»

Apollo Server




Apollo Server

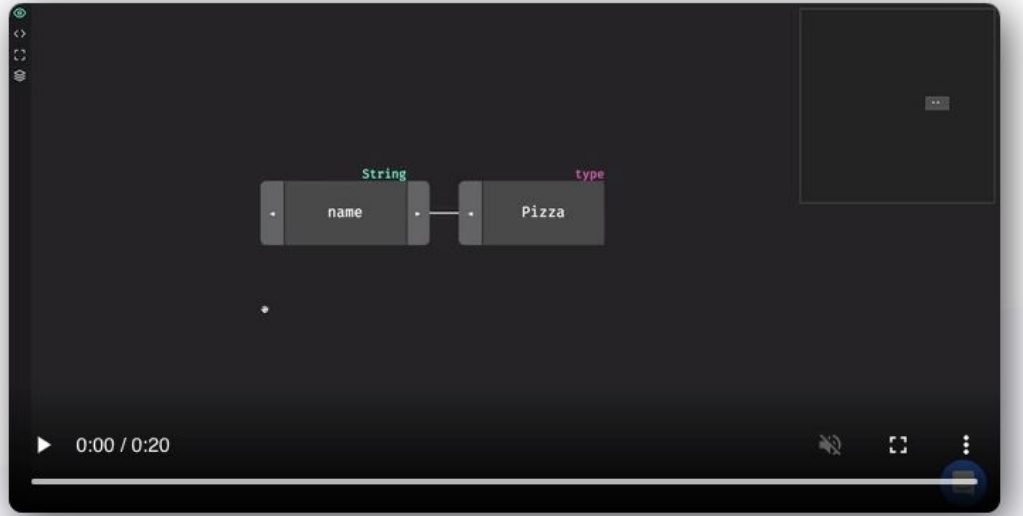
GRAPHQLEDITOR

Star 4,395 GraphQL Blog Changelog 3 Try free

A Better Way to GraphQL

Take your GraphQL schema to the **next level**

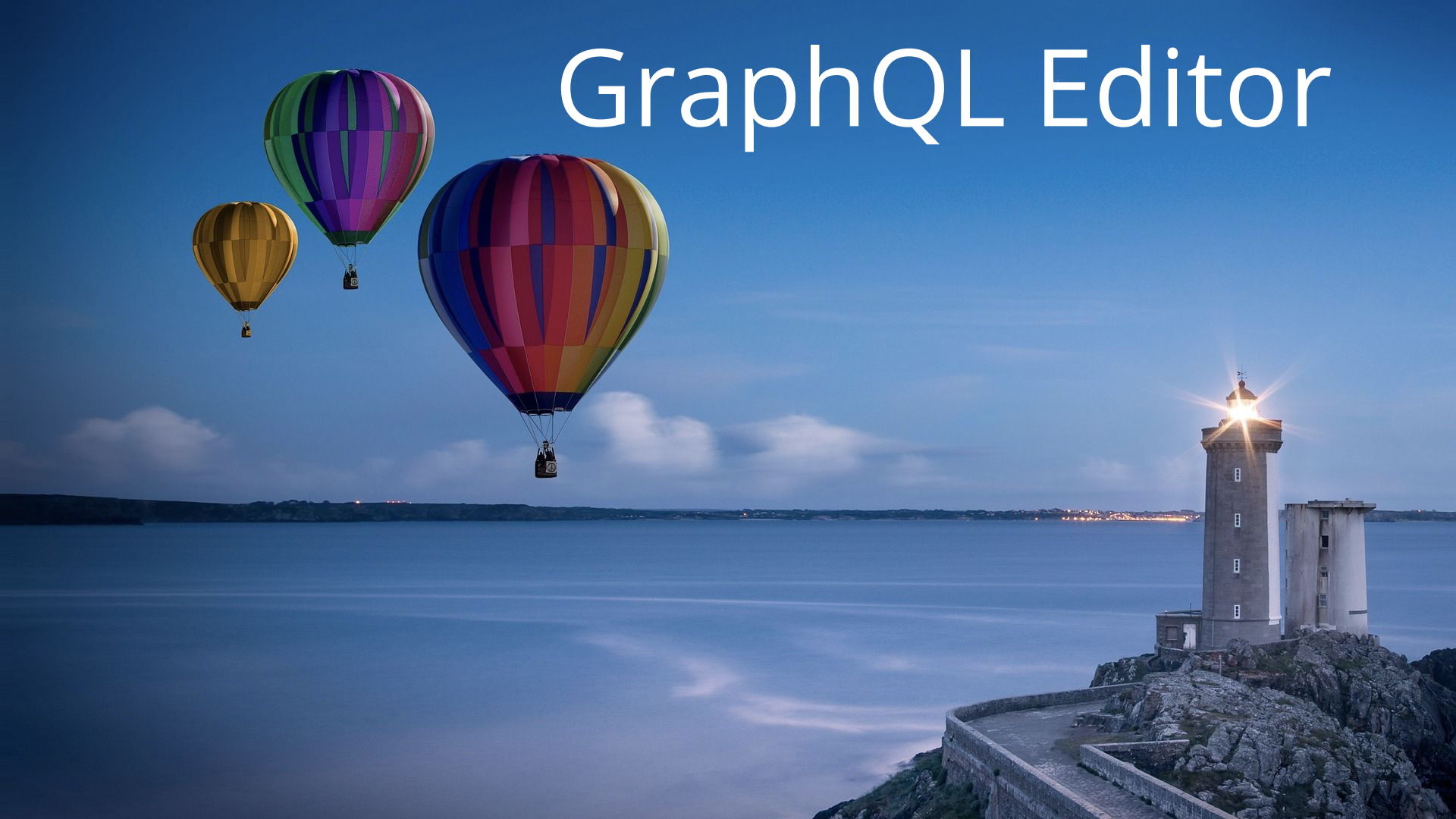
Start now



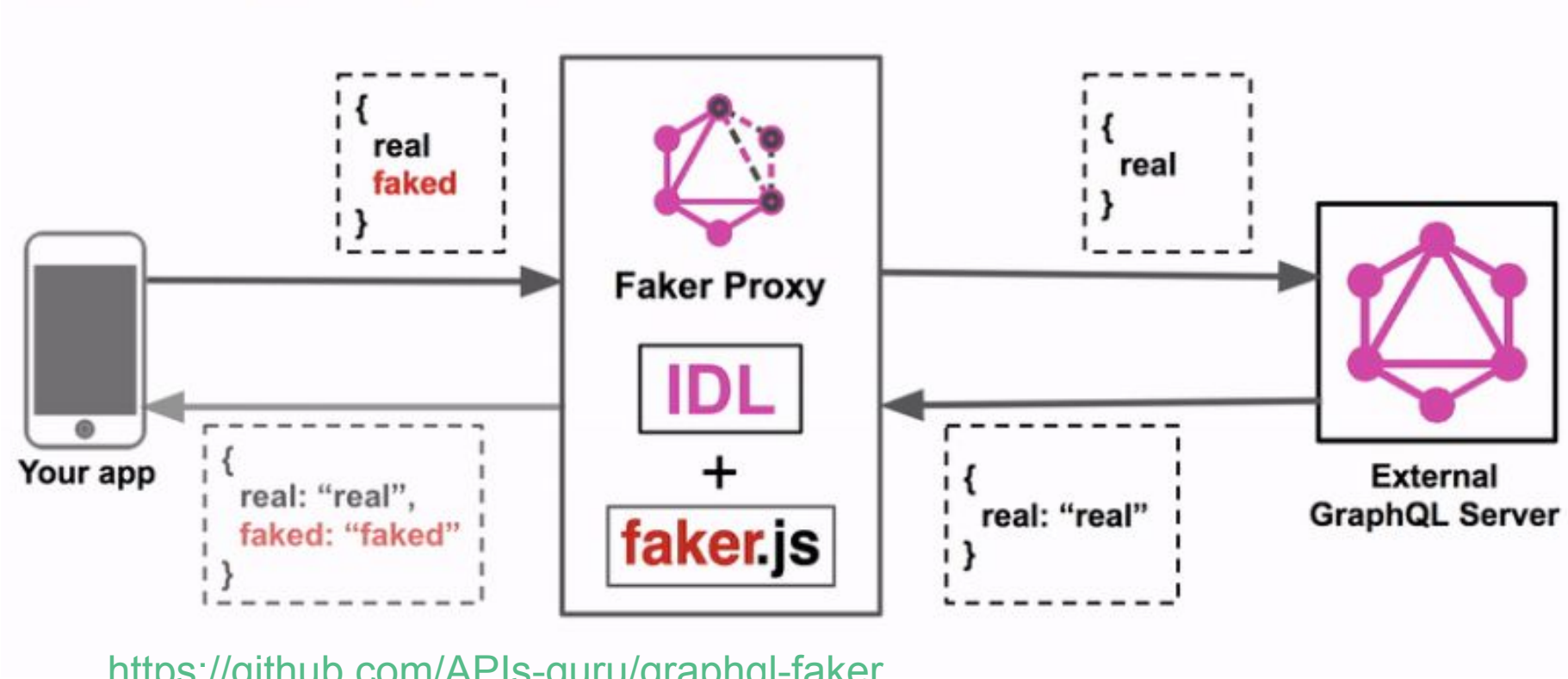
https://graphqleditor.com/


GraphQL
Editor

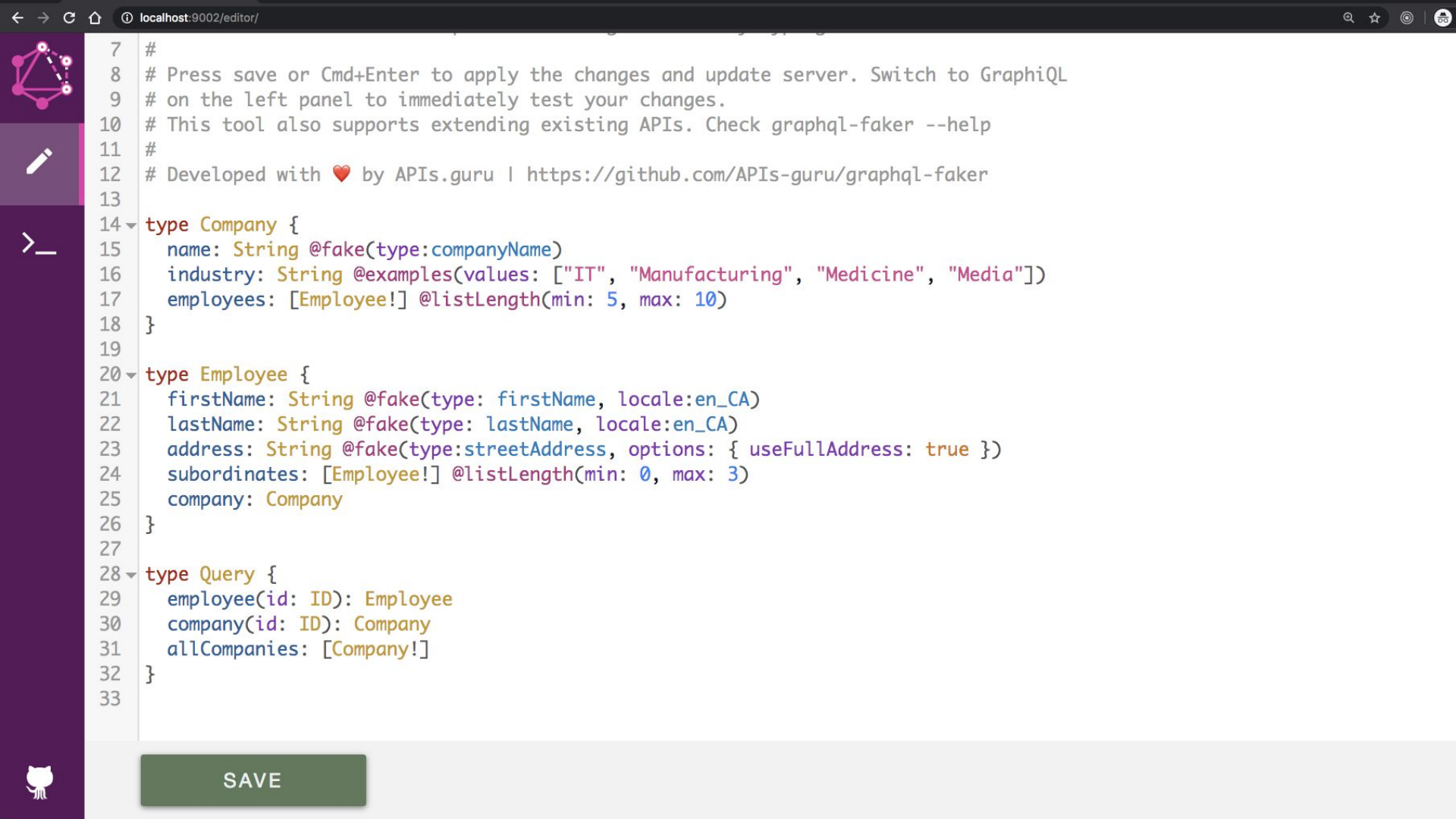
GraphQL Editor



Graphql-faker

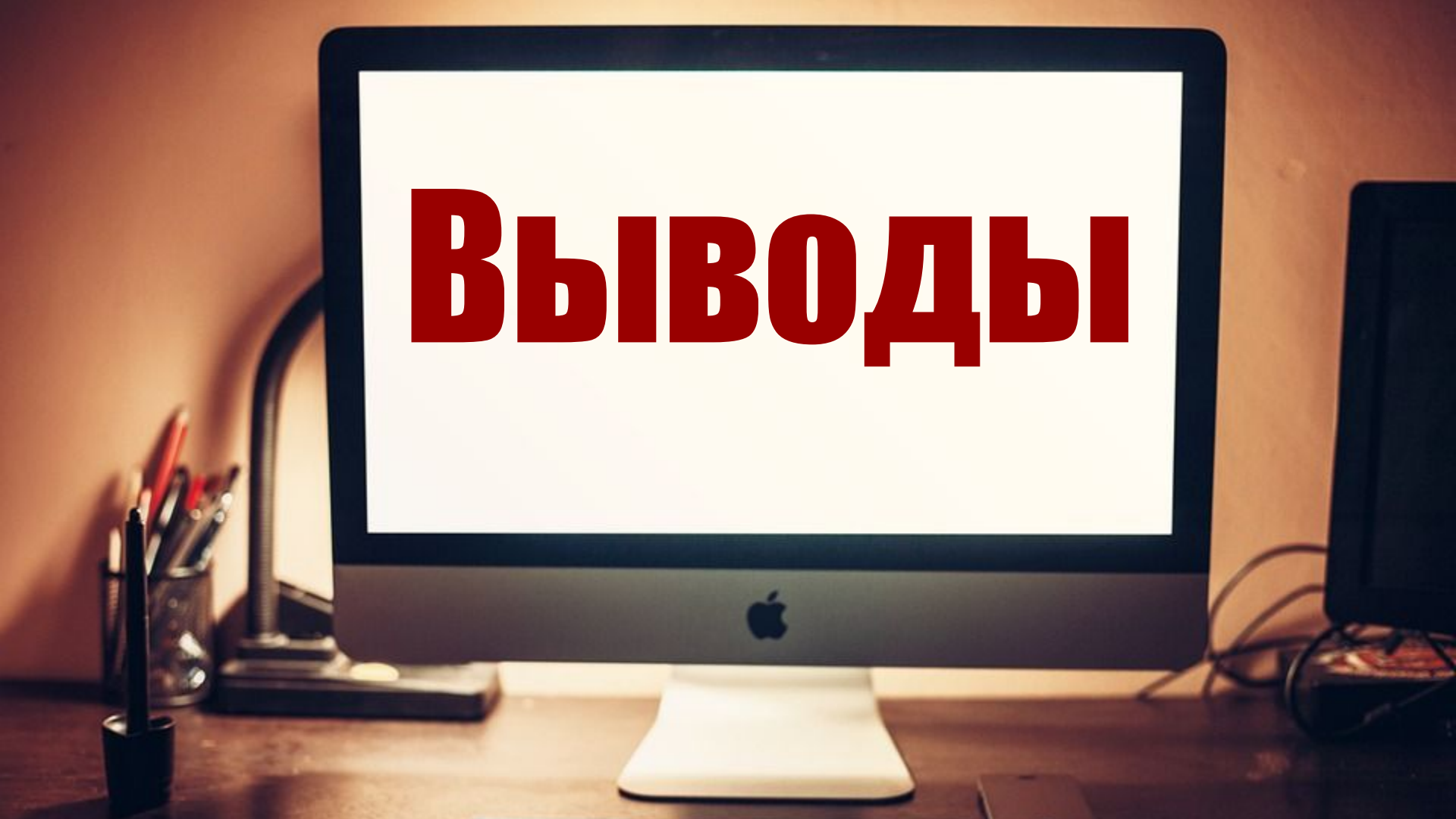


https://github.com/APIs-guru/graphql-faker

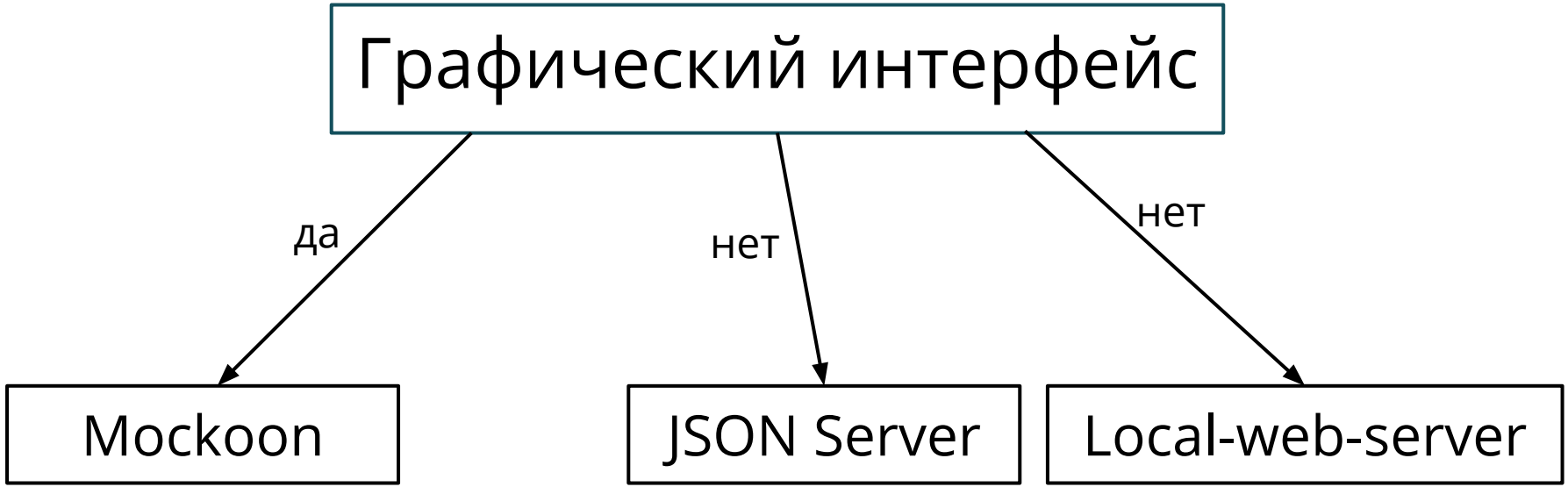

localhost:9002/editor/
#
Press save or Cmd+Enter to apply the changes and update server. Switch to GraphiQL
on the left panel to immediately test your changes.
This tool also supports extending existing APIs. Check graphql-faker --help
#
# Developed with ❤️ by APIs.guru | https://github.com/APIs-guru/graphql-faker
type Company {
name: String @fake(type:companyName)
industry: String @examples(values: ["IT", "Manufacturing", "Medicine", "Media"])
employees: [Employee!] @listLength(min: 5, max: 10)
}
type Employee {
firstName: String @fake(type: firstName, locale:en_CA)
lastName: String @fake(type: lastName, locale:en_CA)
address: String @fake(type:streetAddress, options: { useFullAddress: true })
subordinates: [Employee!] @listLength(min: 0, max: 3)
company: Company
}
type Query {
employee(id: ID): Employee
company(id: ID): Company
allCompanies: [Company!]
}
SAVE

Graphql-faker

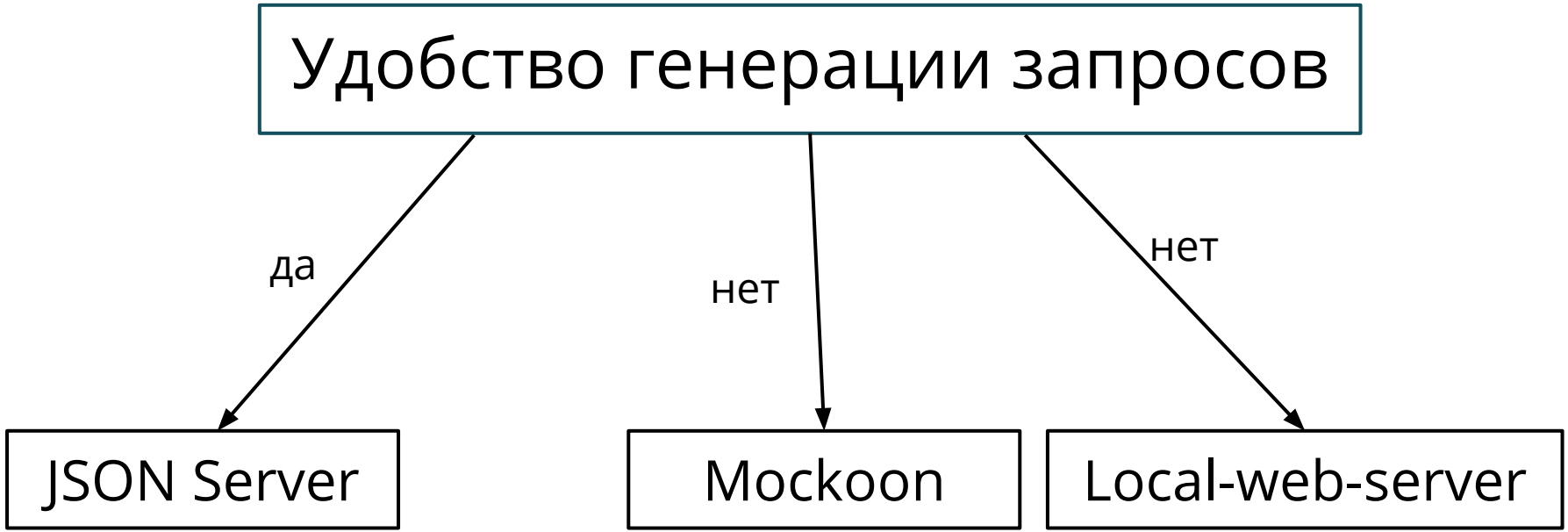
- Более 60 видов различных типов данных
- Множество локалей
- Автозаполнение
- Сохранение обновленной схемы в файл


Выводы

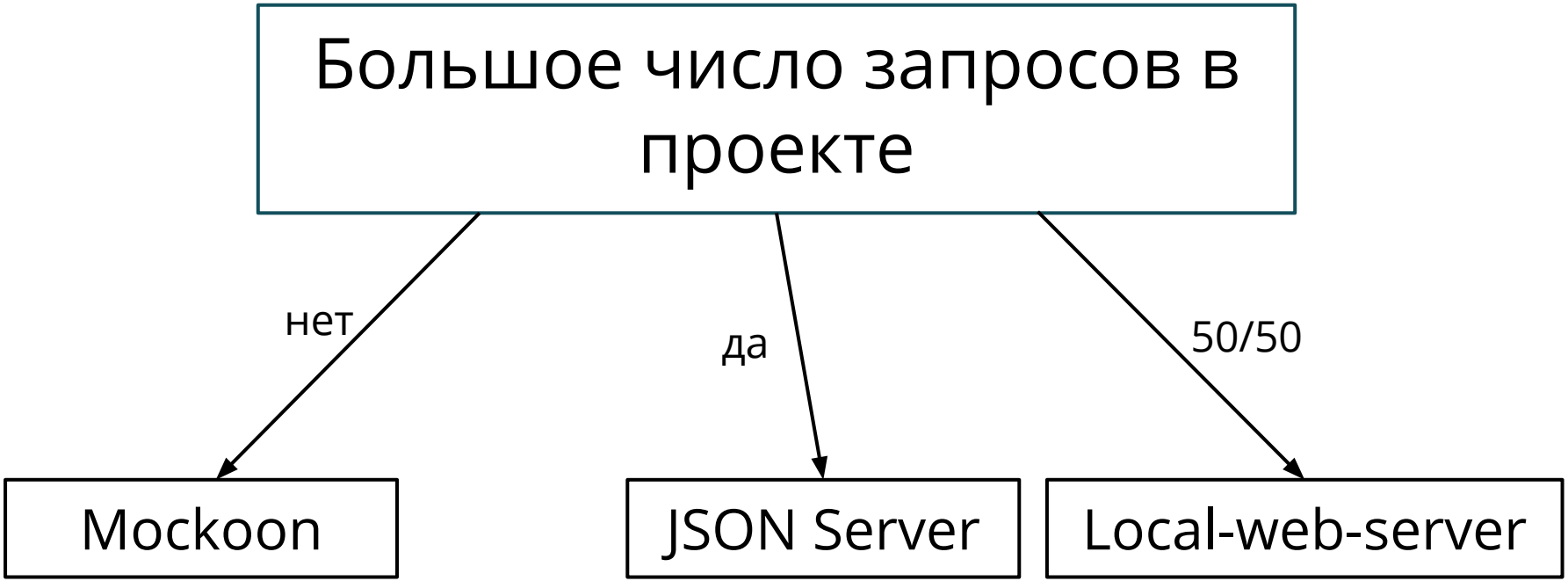
REST API


Графический интерфейс
да
нет
нет
Mockoon
JSON Server
Local-web-server

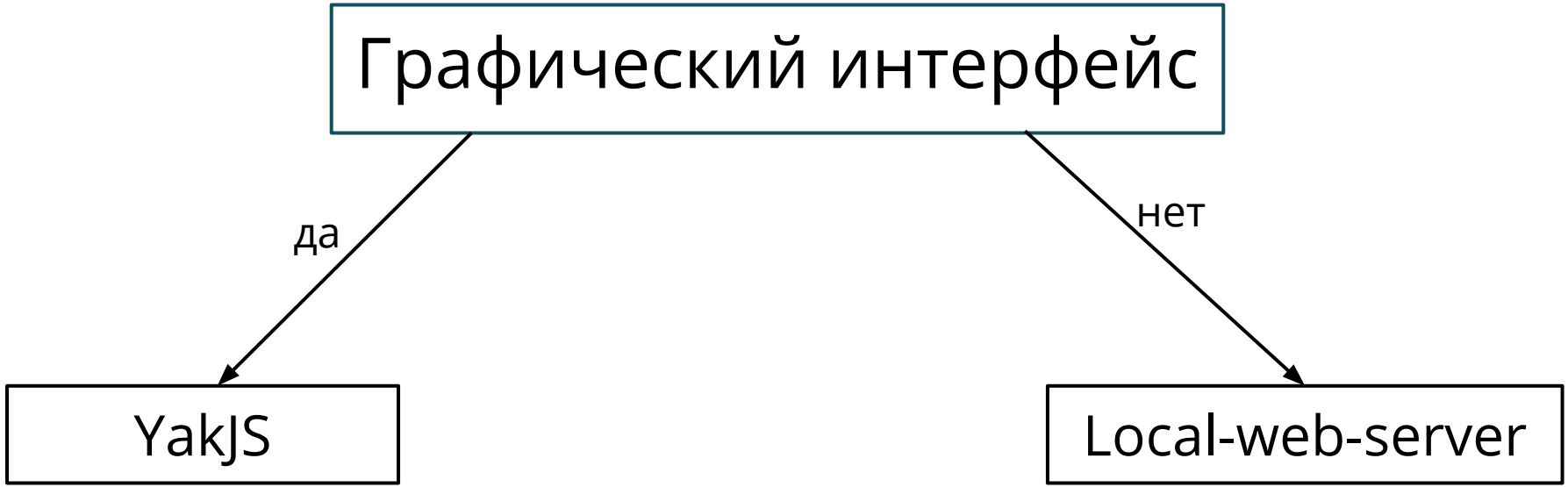
REST API


Удобство генерации запросов
да
нет
нет
JSON Server
Mockoon
Local-web-server

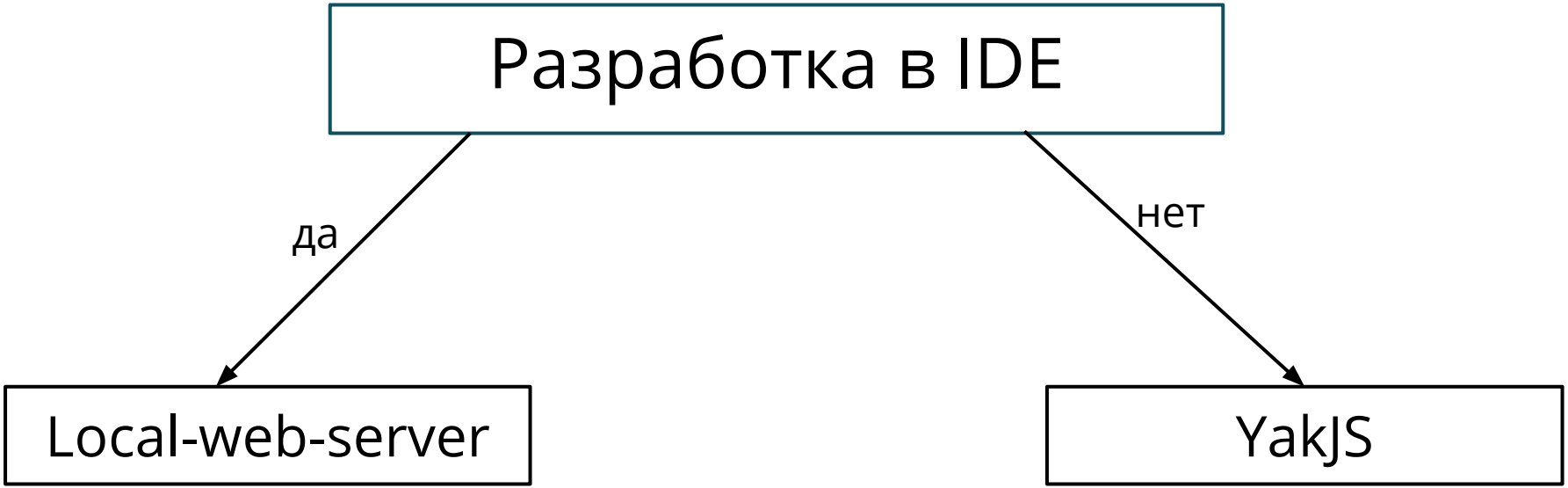
REST API


Большое число запросов в проекте
нет
да
50/50
Mockoon
JSON Server
Local-web-server

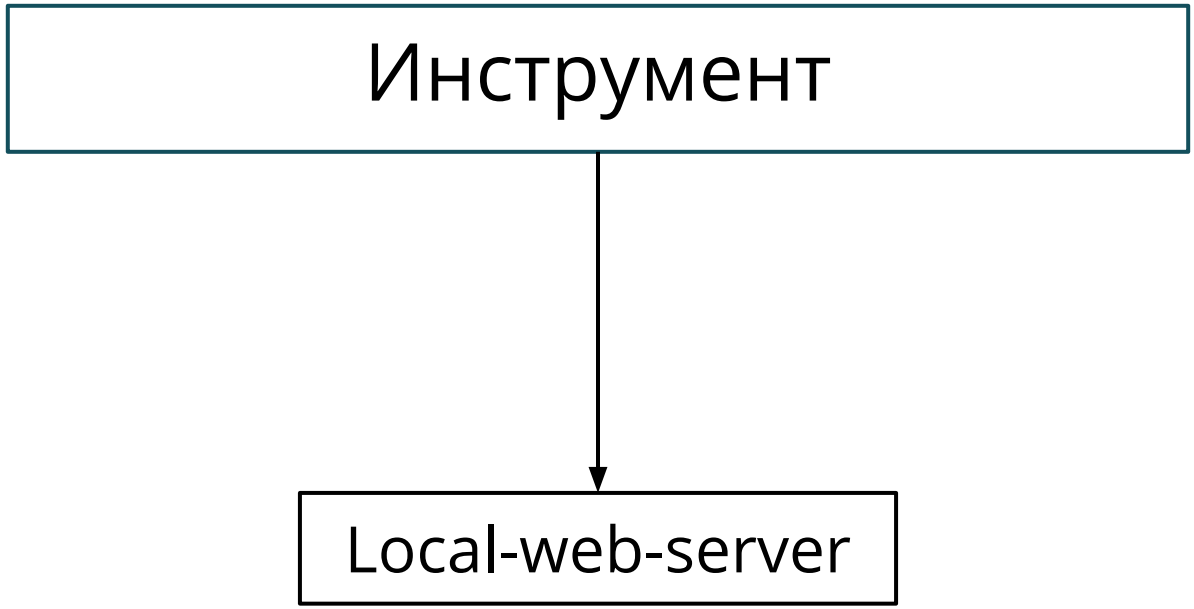
WebSocket



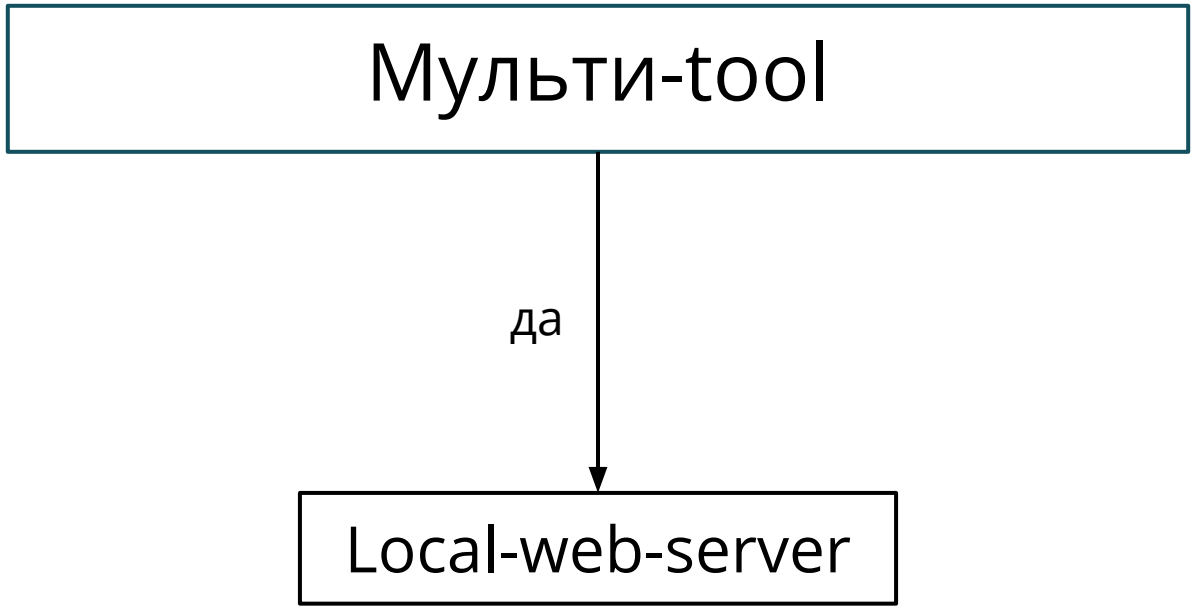
WebSocket



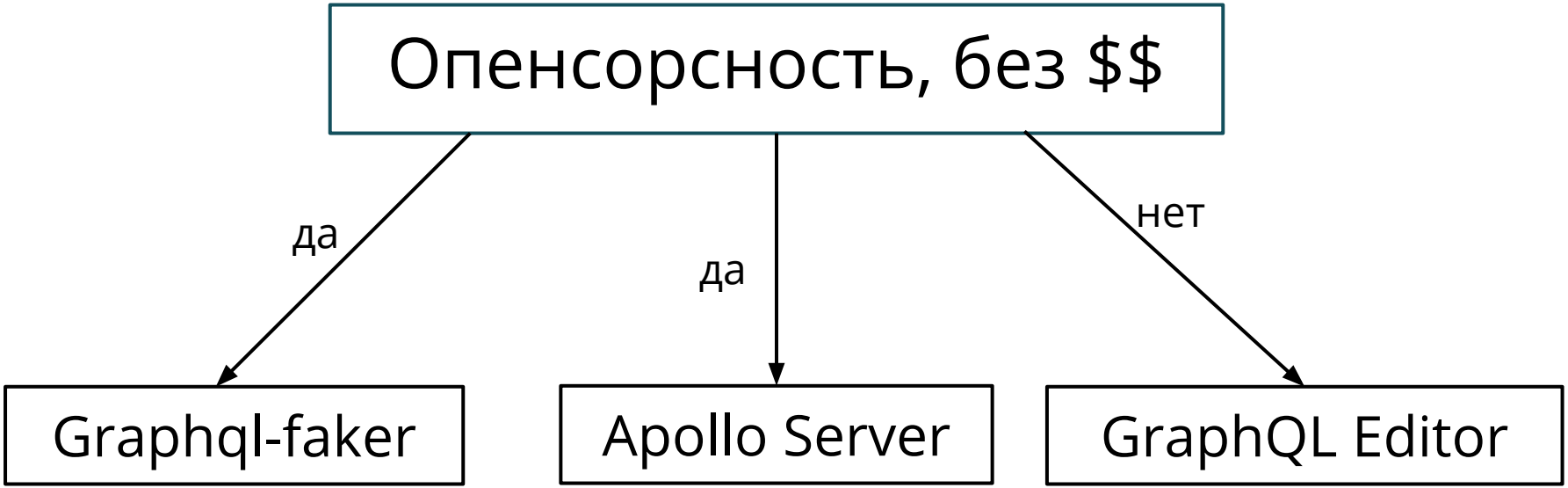
SSE



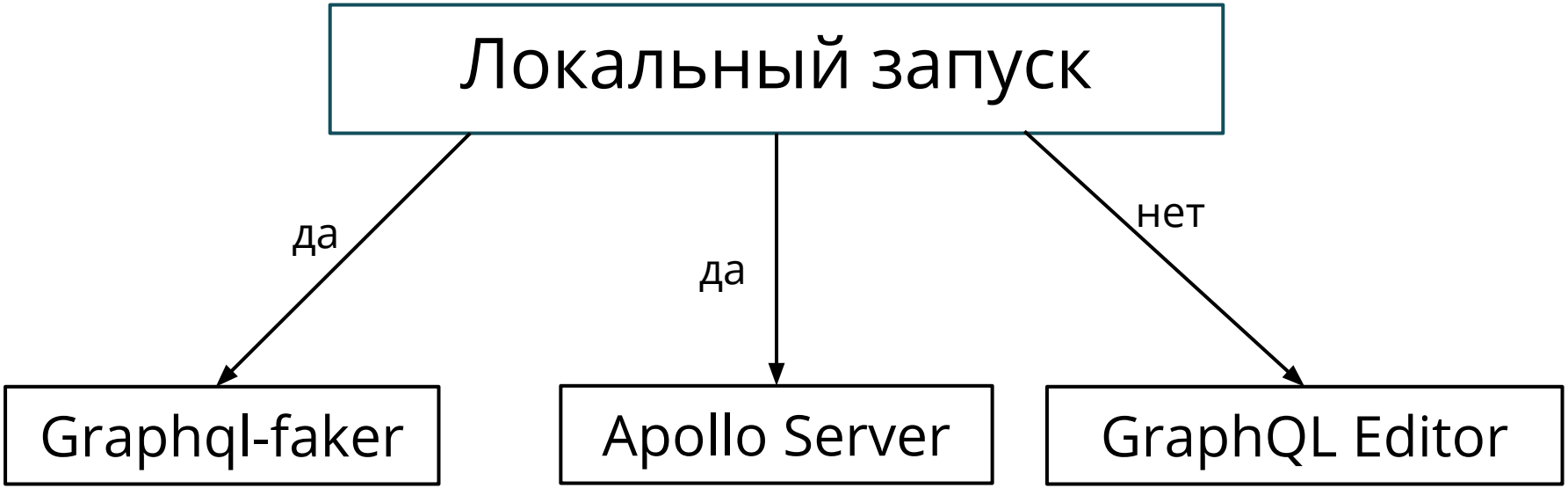
REST API, WebSocket, SSE



GraphQL


Опенсорсность, без $$
да
да
нет
Graphql-faker
Apollo Server
GraphQL Editor

GraphQL


Локальный запуск
да
да
нет
Graphql-faker
Apollo Server
GraphQL Editor

Хотите? ЗаSTUBим!

https://clck.ru/NRygG

https://github.com/Feyweyter/stub-servers

Stubы бэкенда

ЗаSTUBить бэк просто!

Шакшина Мария

Feyweyter

marflenx

marflenx