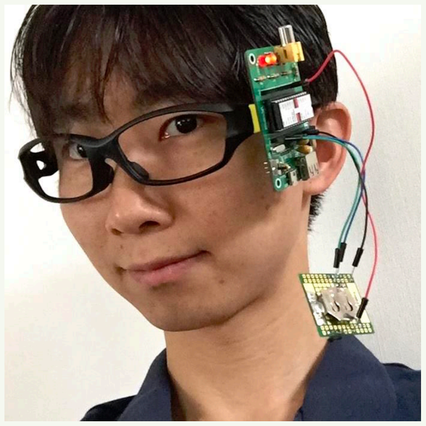
webアプリをつくろう！勉強会
超かんたんフロントｘサーバー編



株式会社 jig.jp 創業者＆会長 福野泰介

@taisukef

webアプリとは？

ブラウザで動くアプリのこと

（PC、iPhone、Android、タブレットなど）

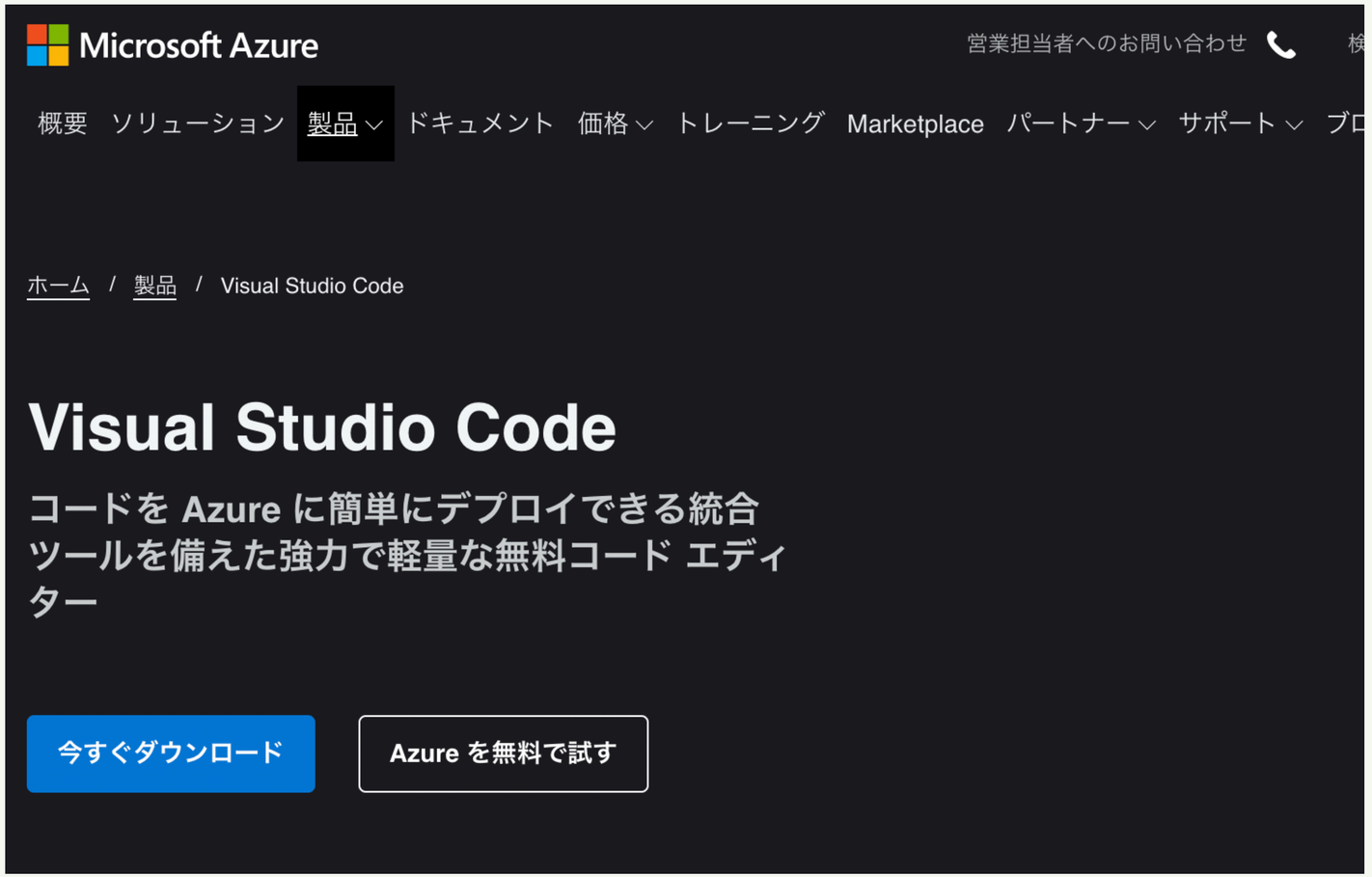
どう作る？

フロントエンド：HTML+CSS+JavaScript で作る
サーバーサイド：サーバー上で動く何かの言語で作る

フロントエンド：HTML+CSS+JavaScript

サーバーサイド：JavaScript (Deno)

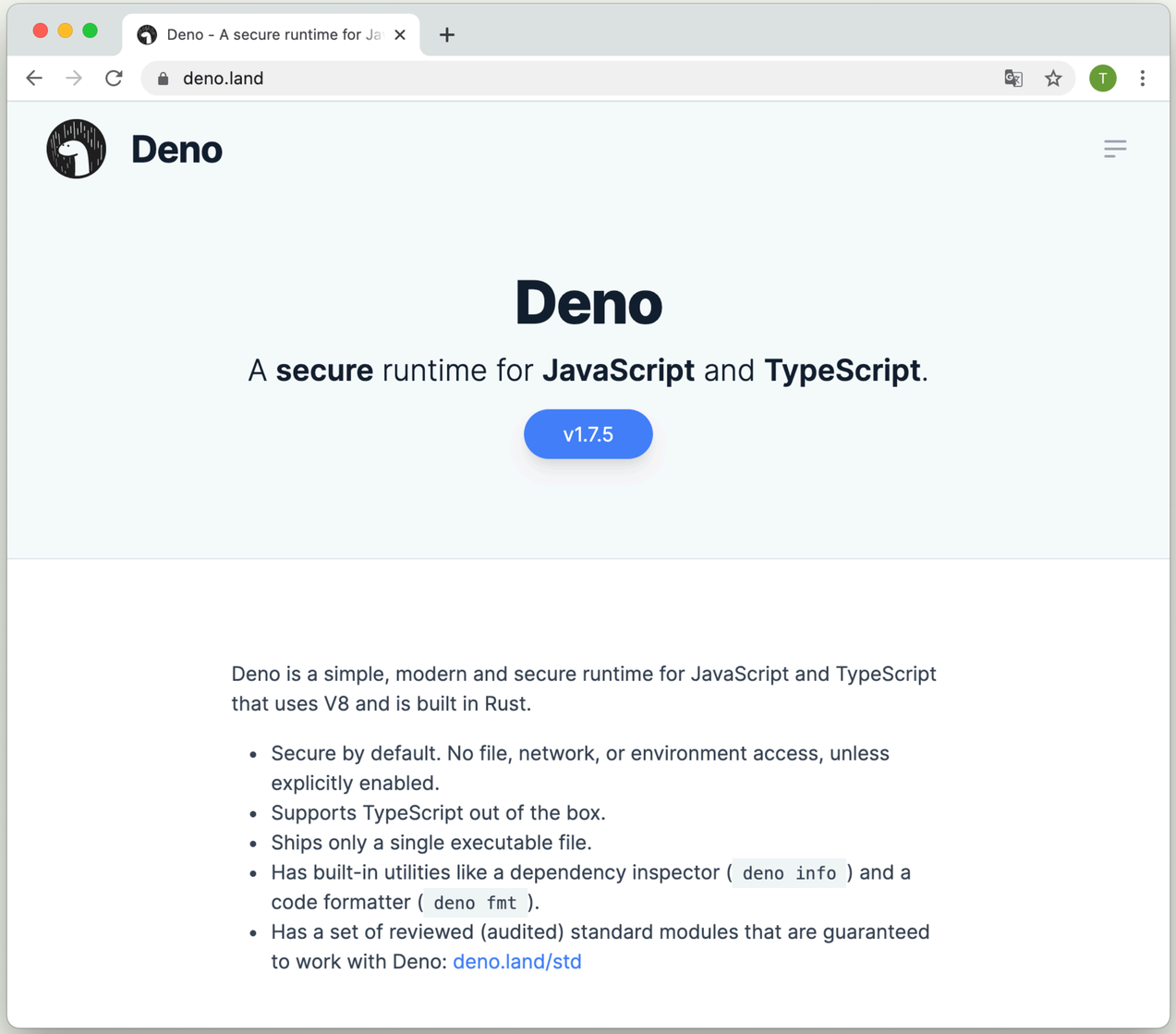
VSCodeを使います



https://azure.microsoft.com/ja-jp/products/visual-studio-code/

1. Denoをインストール
2. サーバーを書く(API)
3. フロントを書く(DOM)

1. Denoをインストール
2. サーバーを書く(API)
3. フロントを書く(DOM)



https://deno.land/

Installation

Deno ships as a single executable with no dependencies. You can install it using the installers below, or download a release binary from the releases page.

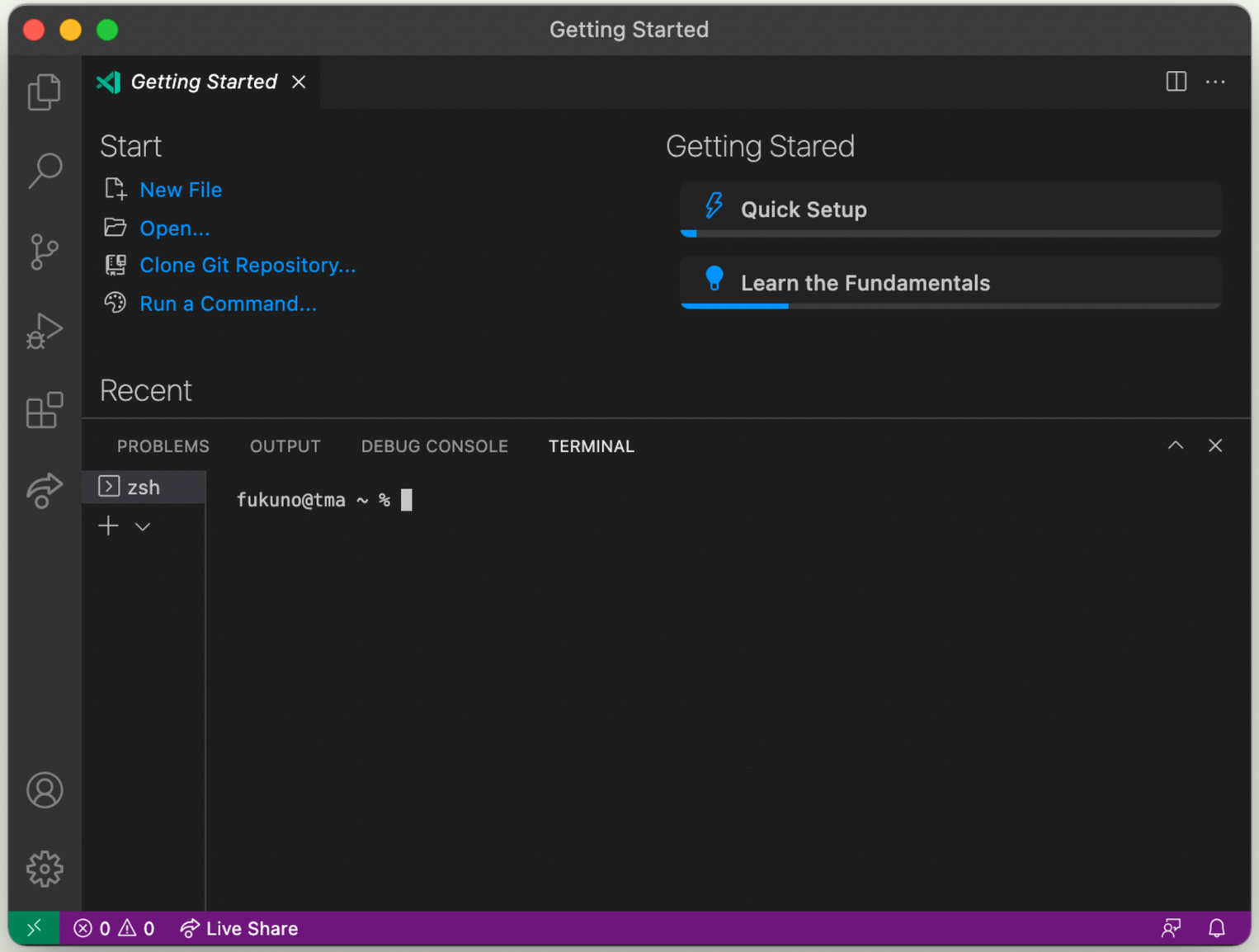
Shell (Mac, Linux):

```
$ curl -fsSL https://deno.land/x/install/install.sh | sh
```

PowerShell (Windows):

```
$ iwr https://deno.land/x/install/install.ps1 -useb | iex
```

インストール用の
スクリプトをコピー



VSCode を開く
メニュー「Terminal」
「New Terminal」
スクリプトを貼り付け
エンター
↓
Denoインストール

Denoインストール終了後、そのままターミナルで

```
$ deno
```

※“$”は環境によって違うプロンプト、打ち込まなくていいよ

```
> 1+1
2
```

```
> prompt()*2
Prompt 16
32
```

止めるときは Ctrl+D

Denoインストールしている人は、upgrade して最新版にしておこう

```
$ deno upgrade
```

1. Denoをインストール
2. サーバーを書く(API)
3. フロントを書く(DOM)

Denoインストール終了後、そのままターミナルで

```
mkdir test1
cd test1
code .
```

ディレクトリ（フォルダ）を作って、 *make directory
そこへ移動して * change directory
VSCode をそのディレクトリ(.)で開く！

server.js というファイルを作成して、下記を打つ

```
import { Server } from "https://js.sabae.cc/Server.js";
new Server(8001);
```

保存する

Terminal、New Terminal をもう一度開き、下記を打つ

```
deno run -A --watch server.js
```

http://localhost:8001 をブラウザで開く、"not found" と表示される

static ディレクトリを作成し、新しいファイル index.html をつくる

```
<h1>Hello!</h1>
```

h1は、HTMLタグの一種で「大見出し」という意味

http://localhost:8001 をブラウザで開く、見える！

server.jsを編集

```
import { Server } from "https://js.sabae.cc/Server.js";
class MyServer extends Server {
    api(path) {
        return path.toUpperCase();
    }
}
new MyServer(8001);
```

保存すると自動的にサーバーが更新される（--watch オプション）

http://localhost:8001/api/test　とかをブラウザで開く

server.jsを編集（オブジェクトを作って返す）

```
import { Server } from "https://js.sabae.cc/Server.js";
class MyServer extends Server {
    api(path) {
        return { name: "jigintern", path: path };
    }
}
new MyServer(8001);
```

http://localhost:8001/api/test　とかをブラウザで開く

```
{"name":"jigintern","path":"/api/test"}
```

↑ JSON です

1. Denoをインストール
2. サーバーを書く(API)
3. フロントを書く(DOM)

testserver に static/index.html を編集

```
<script type=module>
alert("Hi!");
</script>
```

http://localhost:8001 をブラウザで開く、動く！挨拶してくれるよ

testserver に static/index.html を編集

```
<script type=module>
import { fetchJSON } from "https://js.sabae.cc/fetchJSON.js";
window.onload = async () => {
  const data = await fetchJSON("api/test");
  alert(data.name);
};
</script>
```

※ async() => は、awaitを使う引数なしの関数を作る宣言

server.jsを編集

```
import { Server } from "https://js.sabae.cc/Server.js";
const list = [];
class MyServer extends Server {
    api(path, req) {
        console.log(req);
        return { name: "jigintern", path: path };
    }
}
new MyServer(8001);
```

apiでパラメーターを受け取る

console.log で、Terminal上に表示する

testserver に static/index.html を編集

```
<script type=module>
import { fetchJSON } from "https://js.sabae.cc/fetchJSON.js";
window.onload = async () => {
  const data = await fetchJSON("api/test", { name: "jig", age: 17} );
  alert(data.name);
};
</script>
```

※ fetchJSON関数を使ってパラメータをサーバーに渡す

testserver に static/index.html を編集

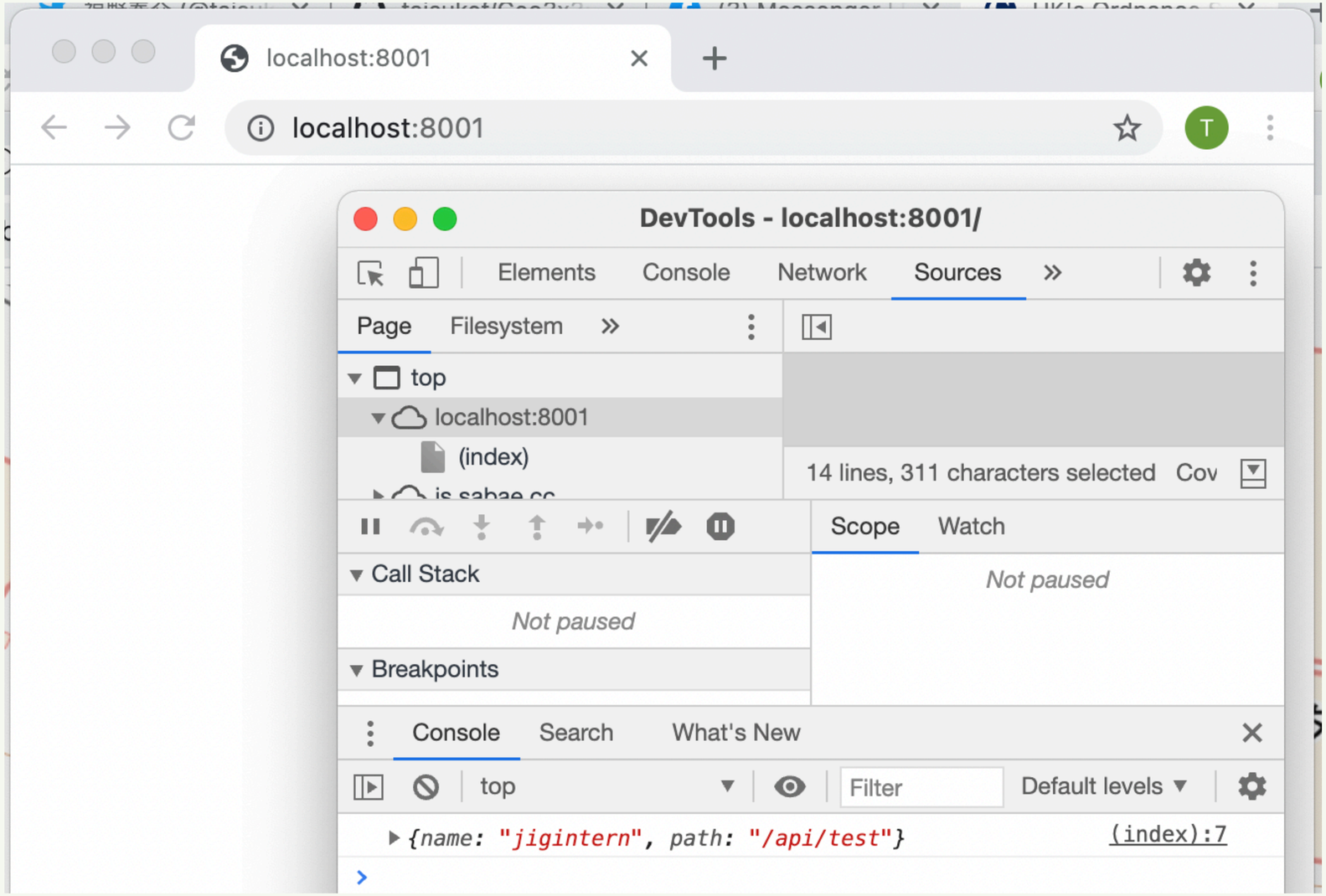
```
<div id=divname></div>
<script type=module>
import { fetchJSON } from "https://js.sabae.cc/fetchJSON.js";
window.onload = async () => {
  const data = await fetchJSON("api/test", { name: "jig", age: 17} );
  divname.textContent = data.name;
};
</script>
```

※divタグにnameというidをつけて、受け取ったデータをセット (DOM操作)

testserver に static/index.html を編集

```
<script type=module>
import { fetchJSON } from "https://js.sabae.cc/fetchJSON.js";
window.onload = async () => {
  const data = await fetchJSON("api/test", { name: "jig", age: 17} );
  alert(data.name);
  console.log(data);
};
</script>
```

※ サーバーでもフロントでも、デバッグに便利な console.log



開発ツールで見てみよう！

Mac: Cmd + Opt + I

Windows: F12

server.jsを編集

```
import { Server } from "https://js.sabae.cc/Server.js";
const list = [];
class MyServer extends Server {
    api(path, req) {
        console.log(req);
        list.push(req);
        return list;
    }
}
new MyServer(8001);
```

掲示板的にサーバーにデータを貯める方法

testserver に static/index.html を編集

```
<script type=module>
import { fetchJSON } from "https://js.sabae.cc/fetchJSON.js";
window.onload = async () => {
  const data = await fetchJSON("api/test", { name: "ab", age: 3 } );
  console.log(data);
};
</script>
```

※ サーバーでもフロントでも、デバッグに便利な console.log

testserver に static/index.html を編集

```
<input id=inp>
<button id=btn>send</button>
<script type=module>
import { fetchJSON } from "https://js.sabae.cc/fetchJSON.js";
btn.onclick = async () => {
  const data = await fetchJSON("api/test", { mes: inp.value } );
  console.log(data);
};
</script>
```

※ サーバーでもフロントでも、デバッグに便利な console.log

掲示板を作ってみよう！

https://fukuno.jig.jp/3169