TypeScript

入門3

Getting Started

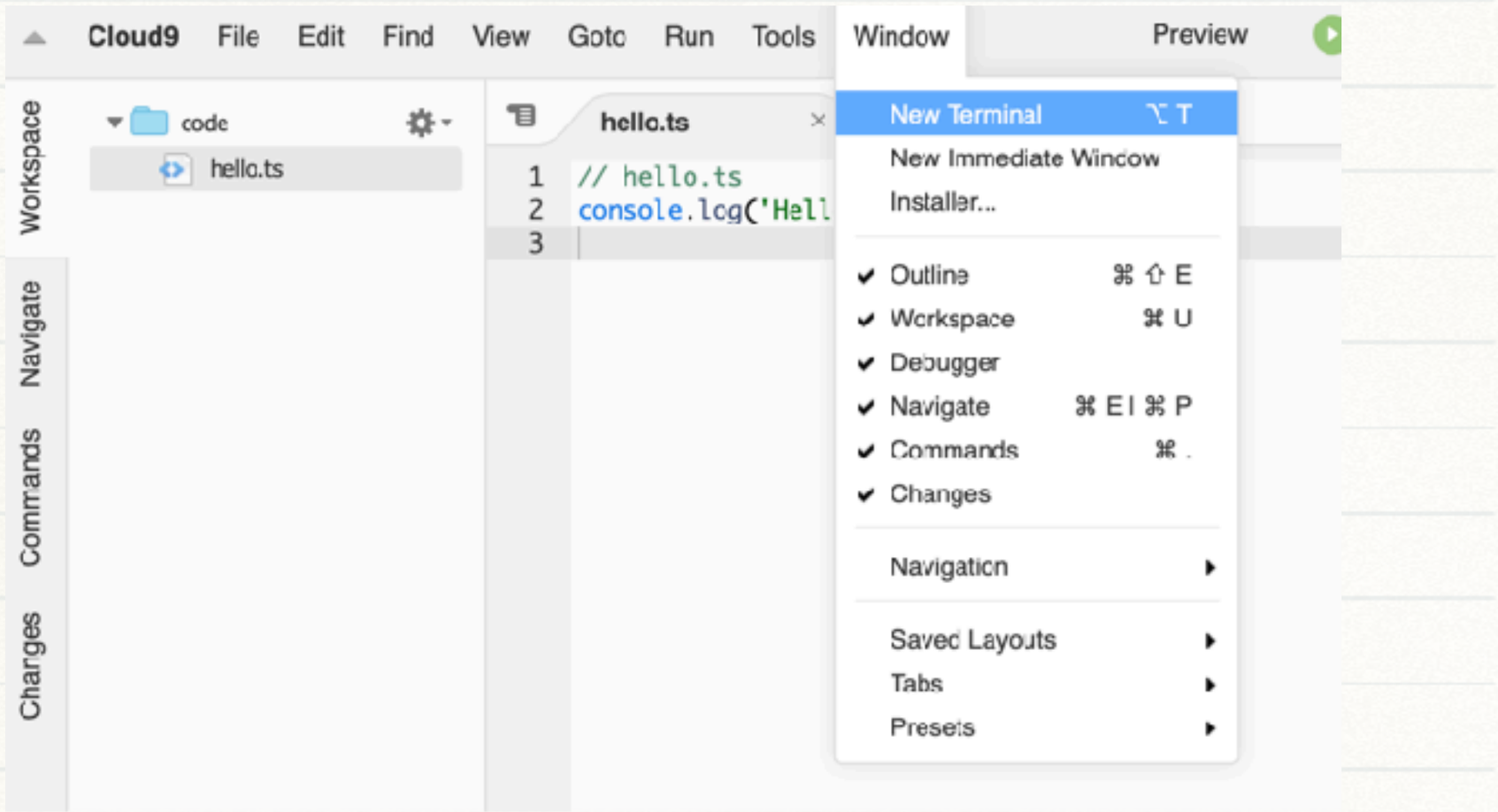
Hello, world!

Hello, world!

hello.ts
拡張子は .ts

```
// hello.ts
console.log('Hello, world!');
```

Window → New Terminal
実行してください



©BLEIZ Ltd. All rights reserved.

deno

10 Things I Regret About Node.js - Ryan Dahl

```
deno run filename.ts
    --help
    --reload
```

```
$ deno run hello.ts
Hello, world!
```

```
ls
pwd
cd

mkdir
rmdir
cp
rm
```

```
$ ls
$ pwd
$ cd
$ pwd

$ cd /code
```

```
zip -r file.zip dir
```

©BLEIZ Ltd. All rights reserved.

Lesson1

interface

JavaScript

```
// ex1.js
function print(p) {
    console.log(p);
}

let p2 = { x: 10, y: 20 };
let p3 = { x: 1, y: 2, z: 3 };

print(p2);
print(p3);

p2 = p3;
print(p2);

p2.z = 0;
```

代入できてしまう

```
$ deno run ex1.js
{ x: 10, y: 20 }
{ x: 1, y: 2, z: 3 }
{ x: 1, y: 2, z: 3 }
```

©BLEIZ Ltd. All rights reserved.

class extends(Object指向)

class

extends

```
// ex2.ts
class Point2 {
    constructor(public x: number, public y: number) { }
}

class Point3 extends Point2 {
    constructor(x: number, y: number, public z: number) {
        super(x, y);
    }
}

function print(p: Point2) {
    console.log(p);
}

let p2 = new Point2(10, 20);
let p3 = new Point3(1, 2, 3);

print(p2);
print(p3);

p2 = p3;
print(p2);

// p2.z = 0;
```

©BLEIZ Ltd. All rights reserved.

interface extends

interface
extends

newなしで使用

```ts
// ex3.ts
interface Point2 {
    x: number;
    y: number;
}

interface Point3 extends Point2 {
    z: number;
}

function print(p: Point2) {
    console.log(p);
}

let p2: Point2 = { x: 10, y: 20 };
let p3 = { x: 1, y: 2, z: 3 } as Point3;

print(p2);
print(p3);

p2 = p3;
print(p2);

// ~~p2.z = 0;~~
```

©BLEIZ Ltd. All rights reserved.

Duck Typing

interface

継承しなくてもOK

```
// ex4.ts
interface Point2 {
    x: number;
    y: number;
}

interface Point3 {
    x: number;
    y: number;
    z: number;
}

function print(p: Point2) {
    console.log(p);
}

let p2: Point2 = { x: 10, y: 20 };
let p3 = { x: 1, y: 2, z: 3 } as Point3;

print(p2);
print(p3);

p2 = p3;
print(p2);

// p2.z = 0;
```

©BLEIZ Ltd. All rights reserved.

ERROR

```
$ deno check ex2.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson1/ex2.ts
error: TS2339 [ERROR]: Property 'z' does not exist on type 'Point2'.
p2.z = 0;
   ^
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson1/ex2.ts:25:4

$ deno check ex3.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson1/ex3.ts
error: TS2339 [ERROR]: Property 'z' does not exist on type 'Point2'.
p2.z = 0;
   ^
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson1/ex3.ts:24:4

$ deno check ex4.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson1/ex4.ts
error: TS2339 [ERROR]: Property 'z' does not exist on type 'Point2'.
p2.z = 0;
   ^
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson1/ex4.ts:26:4
```

©BLEIZ Ltd. All rights reserved.

Lesson2

Type Guard

typeof

```
// ex1.js
function f(x) {
    console.log(x.slice(1));

    if (typeof x === 'string')
        console.log(x.slice(2));
}

f('123');
f(123);
```

```
$ deno run ex1.js
23
3
error: Uncaught TypeError: x.slice is not a function
    console.log(x.slice(1));
                ^
    at f (file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex1.js:3:19)
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex1.js:10:1
```

©BLEIZ Ltd. All rights reserved.

typeof 続き

```ts
// ex1.ts
function f(x: number | string) {
    console.log(x.slice(1));

    if (typeof x === 'string')
        console.log(x.slice(2));
}

f('123');
f(123);
```

```
$ deno check ex1.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex1.ts
error: TS2339 [ERROR]: Property 'slice' does not exist on type 'string | number'.
  Property 'slice' does not exist on type 'number'.
    console.log(x.slice(1));
                      ~~~~~
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex1.ts:3:19
```

©BLEIZ Ltd. All rights reserved.

instanceof

```ts
// ex2.ts
class Foo {
    foo = 123;
}

class Bar {
    bar = 'abc';
}

function f(arg: Foo | Bar) {
    console.log(arg.foo);

    if (arg instanceof Foo)
        console.log(arg.foo);
}

f(new Foo());
f(new Bar());
```

```
$ deno check ex2.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex2.ts
error: TS2339 [ERROR]: Property 'foo' does not exist on type 'Foo | Bar'.
  Property 'foo' does not exist on type 'Bar'.
    console.log(arg.foo);
                    ~~~
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex2.ts:11:21
```

©BLEIZ Ltd. All rights reserved.

in

```ts
// ex3.ts
interface Foo {
    foo: number;
}

interface Bar {
    bar: string;
}

function f(arg: Foo | Bar) {
    console.log(arg.foo);

    if ('foo' in arg)
        console.log(arg.foo);
}

f({ foo: 123 });
f({ bar: 'abc' });
```

```
$ deno check ex3.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex3.ts
error: TS2339 [ERROR]: Property 'foo' does not exist on type 'Foo | Bar'.
  Property 'foo' does not exist on type 'Bar'.
    console.log(arg.foo);
                    ~~~
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex3.ts:11:21
```

©BLEIZ Ltd. All rights reserved.

literal

```ts
// ex4.ts
interface Foo {
    t: 'foo';
    foo: number;
}

interface Bar {
    t: 'bar';
    bar: string;
}

function f(arg: Foo | Bar) {
    console.log(arg.foo);

    if (arg.t === 'foo')
        console.log(arg.foo);
}

f({ t: 'foo', foo: 123 });
f({ t: 'bar', bar: 'abc' });
f({ t: 'boo', bar: 'abc' });
```

©BLEIZ Ltd. All rights reserved.

```
$ deno check ex4.ts
Check file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex4.ts
error: TS2339 [ERROR]: Property 'foo' does not exist on type 'Foo | Bar'.
  Property 'foo' does not exist on type 'Bar'.
    console.log(arg.foo);
                    ~~~
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex4.ts:13:21

TS2322 [ERROR]: Type '"boo"' is not assignable to type '"foo" | "bar"'.
f({ t: 'boo', bar: 'abc' });
    ^
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex4.ts:21:5

    The expected type comes from property 't' which is declared here on type 'Foo | Bar'
        t: 'foo';
        ^
        at file:///Users/shingo1551/Documents/github/course/ts3/lesson2/ex4.ts:3:5

Found 2 errors.
```

©BLEIZ Ltd. All rights reserved.

switch

```ts
// ex5.ts
interface Foo {
    t: 'foo';
    foo: number;
}

interface Bar {
    t: 'bar';
    bar: string;
}

function f(arg: Foo | Bar) {
    switch (arg.t) {
        case 'foo':
            console.log(arg.foo);
            break;

        case 'bar':
            console.log(arg.bar);
            break;
    }
}

f({ t: 'foo', foo: 123 });
f({ t: 'bar', bar: 'abc' });
```

```
$ deno run ex5.ts
123
abc
```

©BLEIZ Ltd. All rights reserved.

Lesson3

例外処理

try ... catch ... finally

```
try {
    ...
} catch (e) {
    ...
} finally {
    ...
}
```

```
// ex1.ts
let s: any;

console.log(s.length);
```

```
// ex2.ts
let s: any;

try {
    console.log('pass 1');
    console.log(s.length)
    console.log('pass 2');
} catch (e) {
    console.log('pass catch');
} finally {
    console.log('pass finally');
}
```

```
$ deno run ex2.ts
pass 1
pass catch
pass finally
```

©BLEIZ Ltd. All rights reserved.

throw

throw *string*

throw Error(*string*)

```
// ex3.ts
function f1() {
    try {
        throw 'Exception';
    } catch (e) {
        console.log(e);
    }
}
f1();

function f2() {
    try {
        throw Error('Exception!!!');
    } catch (e) {
        console.log(e);
    }
}
f2();
```

```
$ deno run ex3.ts
Exception
Error: Exception!!!
    at f2 (file:///Users/shingo1551/Documents/github/course/ts3/lesson3/ex3.ts:13:15)
    at file:///Users/shingo1551/Documents/github/course/ts3/lesson3/ex3.ts:18:1
```

©BLEIZ Ltd. All rights reserved.

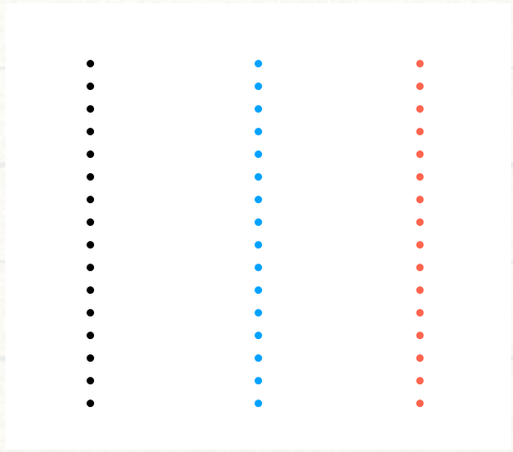
Lesson4

asyc, await

非同期IO

InternetはIOが多い

Multi Thread

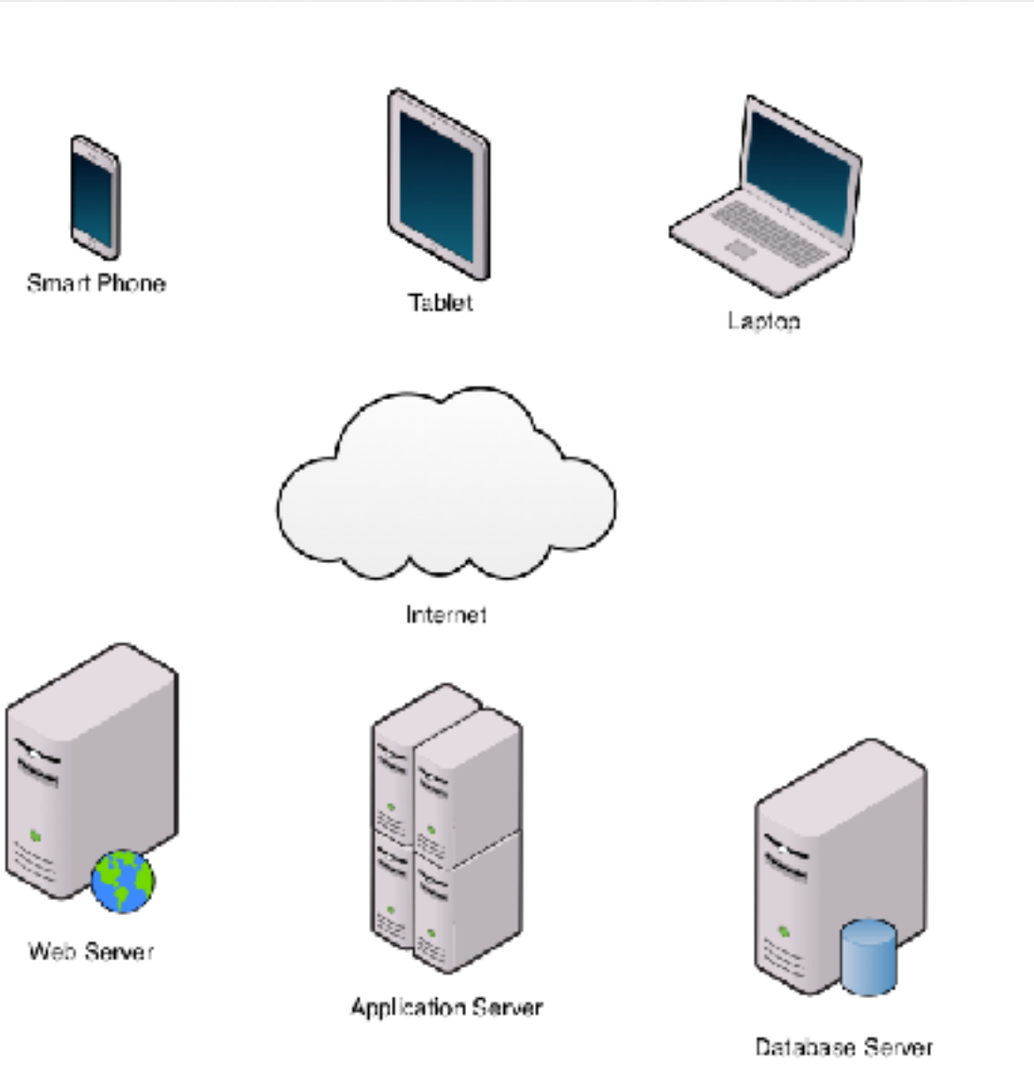


Multi Threadで同時処理を行なうと、

IO待ちによるsleepが長時間発生する

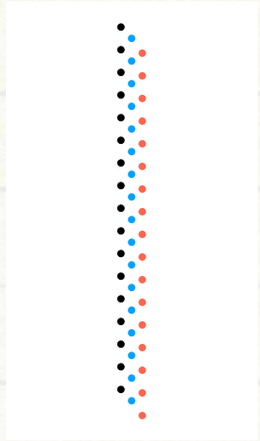
C10K問題

Threadごとに多くのメモリーが割り当てられるため、メモリーが枯渇し処理不能に陥った



IO待ちが発生すると、別の処理を行う

非同期IO



C10K問題

1つのThreadで処理するため、メモリーが枯渇しない

©BLEIZ Ltd. All rights reserved.

async, await

Promise

- resolve
- reject

```
// ex1.ts
export function resolveAfter(sec: number, b: boolean) {
    return new Promise((resolve, reject) => {
        setTimeout(() => {
            if (b)
                resolve('resolved');
            else
                reject('rejected');
        }, sec * 1000);
    });
}
```

async

```
try {
    ...
    await
    ...
} catch (e) {
    ...
}
```

```
// ex2.ts
import { resolveAfter } from './ex1.ts';

export async function asyncCall(sec: number, b: boolean) {
    try {
        console.log('pass 1');
        console.log(await resolveAfter(sec, b));
    } catch (e) {
        console.log('catch:', e);
    }
}
```

©BLEIZ Ltd. All rights reserved.

~~await~~

```ts
// ex3.ts
import { asyncCall } from './ex2.ts';

asyncCall(2, true);
console.log('pass 2');

asyncCall(2, false);
console.log('pass 3');
```

```
$ deno run ex3.ts
pass 1
pass 2
pass 1
pass 3
resolved
catch: rejected
```

©BLEIZ Ltd. All rights reserved.

await

```ts
// ex4.ts
import { asyncCall } from './ex2.ts';

await asyncCall(2, true);
console.log('pass 2');

await asyncCall(2, false);
console.log('pass 3');
```

```
 deno run ex4.ts
pass 1
resolved
pass 2
pass 1
catch: rejected
pass 3
```

©BLEIZ Ltd. All rights reserved.

then

ex2.tsと同等

.then((ok)=>{}, (ng)=>{})

```
// ex5.ts
import { resolveAfter } from './ex1.ts';

export function asyncCall(sec: number, b: boolean) {
    console.log('pass 1');

    return resolveAfter(sec, b).then(
        (ok) => {
            console.log('OK:', ok);
        },
        (ng) => {
            console.log('NG:', ng);
        }
    )
}
```

©BLEIZ Ltd. All rights reserved.

Callback Hell

ex4.tsと同等

*Callback*地獄

```
// ex6.ts
import { asyncCall } from './ex5.ts';

asyncCall(2, true).then(() => {
    console.log('pass 2');

    asyncCall(2, false).then(() => {
        console.log('pass 3');
    });
});
```

```
$ deno run ex6.ts
pass 1
OK: resolved
pass 2
pass 1
NG: rejected
pass 3
```

©BLEIZ Ltd. All rights reserved.

Lesson5

Http Server

Http Server (Standard Library)

```
import { serve } from ...
```

```
// ex1.ts
import { serve } from 'https://deno.land/std@0.148.0/http/server.ts';
serve(() => new Response('Hello World!'), { port: 8080 });
```

```
--allow-net
--watch
```

```
$ deno run --allow-net --watch ex1.ts
```

実行して、ブラウザで表示してみましょう

http://***localhost***:8080/

©BLEIZ Ltd. All rights reserved.

Http Server（続き）

Headers

URL

実行して、ブラウザで表示してみましょう

http://***localhost***:8080/

```
// ex2.ts
import { serve } from 'https://deno.land/std@0.148.0/http/server.ts';

const headers = new Headers([['content-type', 'text/html; charset=UTF-8']]);

const body = `
<html>
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
`;

const handler = (req: Request) => {
  const url = new URL(req.url);
  if (req.method === 'GET' && url.pathname === '/')
    return new Response(body, { headers: headers });
  else
    return new Response('Not Found', { status: 404 });
}

serve(handler, { port: 8080 });
```

```
$ deno run --allow-net ex2.ts
```

©BLEIZ Ltd. All rights reserved.

API Server

```
// ex3.ts
import { serve } from 'https://deno.land/std@0.148.0/http/server.ts';
import { api as member } from './ex4.ts';

const handler = (req: Request) => {
  const pathname = new URL(req.url).pathname;
  if (pathname.startsWith('/member'))
    return member(req, pathname);
  else
    return new Response('Not Found', { status: 404 });
}

serve(handler, { port: 8080 });
```

```
$ deno run --allow-net ex3.ts
```

実行して、ブラウザで表示してみましょう

http://***localhost***:8080/member/id-001

©BLEIZ Ltd. All rights reserved.

API Server (続き)

```
// ex4.ts
import * as db from './ex5.ts';

export function api(req: Request, pathname: string) {
  try {
    if (req.method === 'GET')
      return get(pathname);
    else
      return new Response('Not Found', { status: 404 });
  } catch (e) {
    return new Response(e, { status: 400 });
  }
}

const headers = new Headers([['content-type', 'application/json']]);

function get(pathname: string) {
  const body = { status: 0, member: db.get(parseUrl(pathname)) };
  return new Response(JSON.stringify(body), { headers });
}

function parseUrl(pathname: string) {
  const params = pathname.split('/');
  if (params.length !== 3 || !params[2])
    throw 'Bad Request';
  else
    return params[2];
}
```

©BLEIZ Ltd. All rights reserved.

```
[index: string]: Member;
```

```
// ex5.ts
export interface Member {
  name: string;
  age: number;
}

interface MemberDB {
  [index: string]: Member;
}

const memberdb: MemberDB = {
  'id-001': { name: 'suzuki', age: 30 },
  'id-002': { name: 'tanaka', age: 40 },
};

export function get(id: string) {
  return memberdb[id];
}
```

©BLEIZ Ltd. All rights reserved.

Lesson6

Http Client

fetch

fetch ... then

res.json()

```
// ex1.ts
fetch('http://localhost:8080/member/id-001')
    .then((res) => {
        return res.json();
    }).then((obj) => {
        console.log(obj);
    });

fetch('http://localhost:8080/member/id-002')
    .then(res => res.json()).then(obj => console.log(obj));

try {
    const res = await fetch('http://localhost:8080/member/id-002');
    console.log(await res.json());
} catch (e) {
    console.log(e);
}
```

```
$ deno run --allow-net ex1.ts
{ status: 0, member: { name: "tanaka", age: 40 } }
{ status: 0, member: { name: "tanaka", age: 40 } }
{ status: 0, member: { name: "suzuki", age: 30 } }
```

©BLEIZ Ltd. All rights reserved.

post

```
// ex2.ts
import { Member } from './ex5.ts';

const member: Member = {
    name: 'yamada',
    age: 45
}

const headers = new Headers([['content-type', 'application/json']]);

const opt = {
    method: 'POST',
    headers: headers,
    body: JSON.stringify(member)
};

try {
    const res = await fetch('http://localhost:8080/member/id-003', opt);
    console.log(await res.json());
} catch (e) {
    console.log(e);
}
```

```
$ deno run --allow-net ex2.ts
{ status: 0 }
```

©BLEIZ Ltd. All rights reserved.

API Server

```
// ex3.ts
import { serve } from 'https://deno.land/std@0.148.0/http/server.ts';
import { api as member } from './ex4.ts';

const handler = (req: Request) => {
  const pathname = new URL(req.url).pathname;
 if (pathname.startsWith('/member'))
    return member(req, pathname);
  else
    return new Response('Not Found', { status: 404 });
}

serve(handler, { port: 8080 });
```

©BLEIZ Ltd. All rights reserved.

API Server (続き)

```
// ex4.ts
import * as db from './ex5.ts';

export function api(req: Request, pathname: string) {
  try {
    if (req.method === 'GET')
      return get(pathname);
    else if (req.method === 'POST')
      return post(req, pathname);
    else
      return new Response('Not Found', { status: 404 });
  } catch (e) {
    return new Response(e, { status: 400 });
  }
}

    ...

async function post(req: Request, pathname: string) {
  const text = await req.text();
  db.set(parseUrl(pathname), JSON.parse(text));
  return new Response(JSON.stringify({ status: 0 }), { headers });
}
```

```
await req.text();
```

©BLEIZ Ltd. All rights reserved.

API Server (続き2)

```
// ex5.ts
export interface Member {
  name: string;
  age: number;
}

interface MemberDB {
  [index: string]: Member;
}

const memberdb: MemberDB = {
  'id-001': { name: 'suzuki', age: 30 },
  'id-002': { name: 'tanaka', age: 40 },
};

export function get(id: string) {
  return memberdb[id];
}

export function set(id: string, member: Member) {
  memberdb[id] = member;
}

export function del(id: string) {
  delete memberdb[id];
}
```

delete

©BLEIZ Ltd. All rights reserved.

Question

ex4.tsに**DEL**を追加してください

DELを使用するclientを作成してください

©BLEIZ Ltd. All rights reserved.

Answer

```
// ext4.ts

    省略

export function api(req: Request, pathname: string) {
  try {
    if (req.method === 'GET')
      return get(pathname);
    else if (req.method === 'POST')
      return post(req, pathname);
    else if (req.method === 'DEL')
      return del(pathname);
    else
      return new Response('Not Found', { status: 404 });
  } catch (e) {
    return new Response(e, { status: 400 });
  }
}

    省略

function del(pathname: string) {
  db.del(parseUrl(pathname));
  return new Response(JSON.stringify({ status: 0 }), { headers });
}
```

©BLEIZ Ltd. All rights reserved.

```
// ans2.ts
const opt = {
    method: 'DEL'
};

try {
    const res = await fetch('http://localhost:8080/member/id-001', opt);
    console.log(await res.json());
} catch (e) {
    console.log(e);
}
```

```
$ deno run --allow-net ans2.ts
{ status: 0 }
```

©BLEIZ Ltd. All rights reserved.

Appendix

Http Client

Runtime API

- Deno.readFileSync
- Deno.args

res.status

```
// fetch.ts
import { Request, Result } from './interface.ts';

const headers = new Headers([['content-type', 'application/json']]);
const results = [] as Result[];

const decoder = new TextDecoder("utf-8");
const data = Deno.readFileSync(Deno.args[0]);
const scenario = JSON.parse(decoder.decode(data)) as Request[];

for (const req of scenario) {
    const opt = {
        method: req.method,
        headers: headers,
        body: req.member ? JSON.stringify(req.member) : undefined
    };

    try {
        const res = await fetch(`http://localhost:8080/member/${req.id}`, opt);
        if (res.status !== 200)
            results.push({ req: req, err: "" + res.status });
        else
            results.push({ req: req, res: await res.json() });
    }
    catch (e) {
        console.error(e);
        results.push({ req: req, err: '' + e });
    }
}

console.log(results);
```

©BLEIZ Ltd. All rights reserved.

```ts
// interface.ts
export interface Member {
    name: string;
    age: number;
}

export interface Request {
    method: 'GET' | 'POST' | 'DEL';
    id: string;
    member?: Member;
}

export interface Response {
    status: number;
    member?: Member;
}

export interface Result {
    req: Request;
    res?: Response;
    err?: string;
}
```

©BLEIZ Ltd. All rights reserved.

scenario.json

```
[
  {
    "method": "GET",
    "id": "id-001"
  },
  {
    "method": "POST",
    "id": "id-002",
    "member": {
      "name": "nakamura",
      "age": 19
    }
  },
  {
    "method": "PUT",
    "id": "id-003",
    "member": {
      "name": "nakamura",
      "age": 19
    }
  },
  {
    "method": "DEL",
    "id": "id-002"
  }
]
```

©BLEIZ Ltd. All rights reserved.

--allow-read

scenario.json -> Deno.args[0]

```
$ deno run --allow-read --allow-net fetch.ts scenario.json
[
  {
    req: { method: "GET", id: "id-001" },
    res: { status: 0, member: { name: "suzuki", age: 30 } }
  },
  {
    req: { method: "POST", id: "id-002", member: { name: "nakamura", age: 19 } },
    res: { status: 0 }
  },
  {
    req: { method: "PUT", id: "id-003", member: { name: "nakamura", age: 19 } },
    err: "404"
  },
  { req: { method: "DEL", id: "id-002" }, res: { status: 0 } }
]
```

©BLEIZ Ltd. All rights reserved.

Link

TypeScript	https://www.typescriptlang.org
deno	https://deno.land
10 Things I Regret About Node.js	https://youtu.be/M3BM9TB-8yA
Visual Studio Code (VScode)	https://code.visualstudio.com

©BLEIZ Ltd. All rights reserved.