



TypeScript

入門1

Getting Started

Hello, world!

Hello, world!

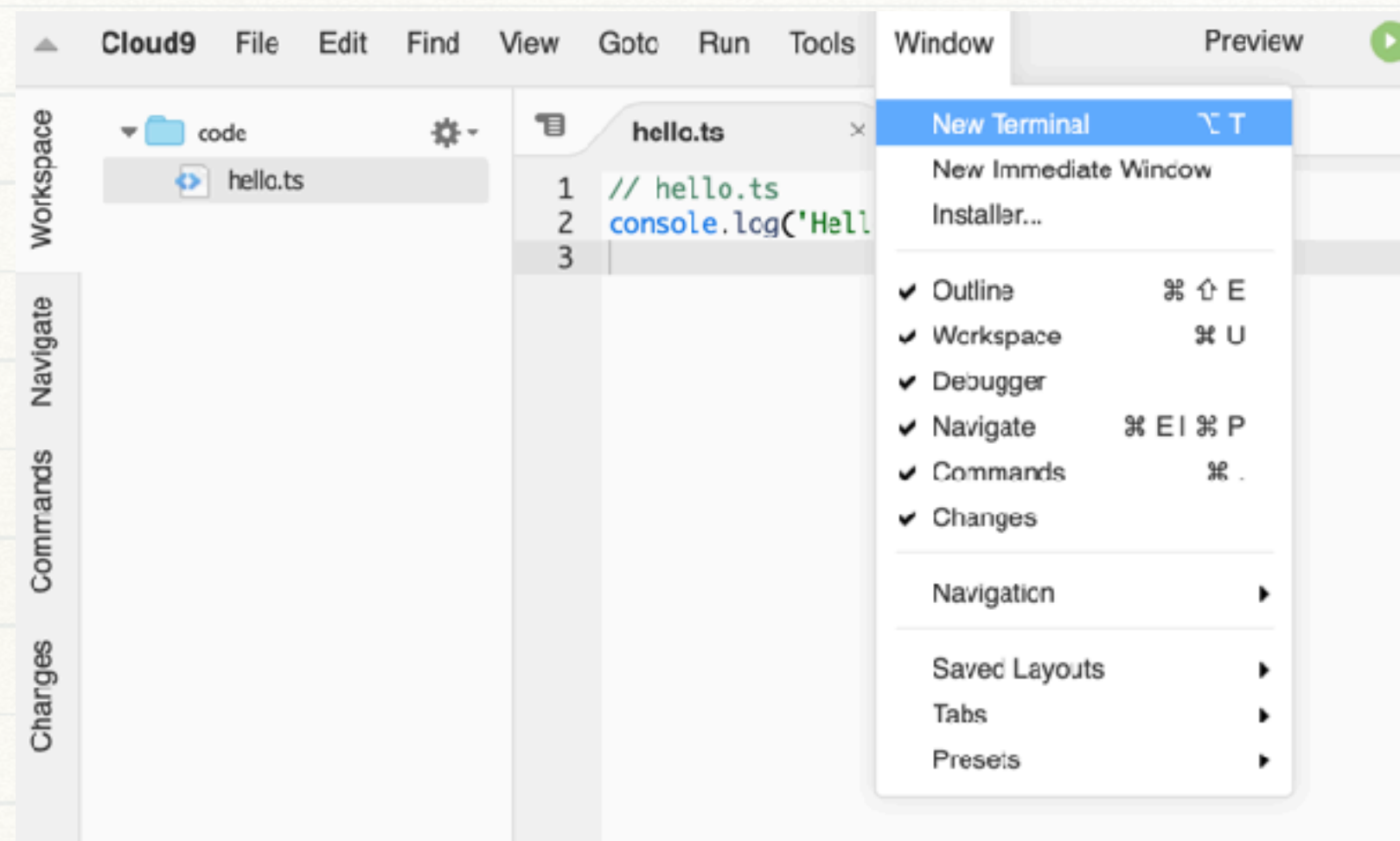
hello.ts

拡張子は **.ts**

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

Window → New Terminal

実行してください



10 Things I Regret About Node.js - Ryan Dahl

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

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

拡張子 **.js** も試しましょう

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

```
ls  
pwd  
cd
```

```
$ ls  
$ pwd  
$ cd  
$ pwd
```

```
mkdir  
rmdir  
cp  
rm
```

```
$ cd /code
```

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


コメント

```
// コメント  
console.log('Hello, world!');    // 行の途中からコメント  
  
/*  
    複数行のコメント  
*/  
  
/**  
 * document comment  
 */
```

TYPEDOC

<https://typedoc.org/>

<https://typedoc.org/guides/doccomments/>

Lesson1

リテラル、変数、型

リテラル 数値、文字列

数値 (**number**)

浮動小数点、**整数はない**

```
1
10.1
1.23e-4

0b1111
0o10
0x12

12345678901234567890 // -> 12345678901234567000
```

bigint

```
12345678901234567890n // bigint
1234567890123456789012345678901234567890123456789001234567890n
```

文字列 (**string**)

環境によっては \ ではなく **¥** を使用

MacOSでは、キーボードの設定を **U.S.** にして \ を使用してください

```
'string'
"string"
'''
'''

'\ '
"\"

'\\'
'\n'
```

リテラル 真偽値、Object、配列

真偽値

```
true  
false
```

Object {}

```
{ x: 1.1, y: 3, z: 4.4 }  
  
{  
  name: 'yamada',  
  age: 28  
}
```

配列Array []

```
[ 1.1, 3, 4.4 ]  
  
[  
  [ 1, 2, 3 ],  
  [ 4, 5, 6 ],  
  [ 7, 8, 9 ]  
]  
  
[  
  { name: 'yamada', age: 28 },  
  { name: 'tanaka', age: 34 }  
]
```


変数 `const`, `let`, `var`

なるべく、`const` を使用

`let` は値を変更できる

なるべく、`var` は使用しないこと

```
// ex1.ts
const a = 1;
a = 2;

let b = 1;
b = 2;
b = 'abc';
```

```
$ deno run ex1.ts
error: Uncaught TypeError: Assignment to constant variable.
a = 2;
^
    at file:///Users/shingo1551/Documents/github/course/ts1/lesson1/ex1.ts:3:2
```

```
$ deno check ex1.ts
Check file:///Users/shingo1551/Documents/github/course/ts1/lesson1/ex1.ts
error: TS2588 [ERROR]: Cannot assign to 'a' because it is a constant.
a = 2;
^
    at file:///Users/shingo1551/Documents/github/course/ts1/lesson1/ex1.ts:3:1

TS2322 [ERROR]: Type 'string' is not assignable to type 'number'.
b = 'abc';
^
    at file:///Users/shingo1551/Documents/github/course/ts1/lesson1/ex1.ts:7:1
Found 2 errors.
```

拡張子 `.js` も試しましょう

型 typeof

さまざまな、型

number

string

boolean

object

function

undefined

bigint

symbol

型 typeof number, bigint

```
// ex2.ts
const log = console.log;

// number
log(typeof 1, 1);
log(typeof 10.1, 10.1);
log(typeof 1.23e-4, 1.23e-4);

log(typeof 0b1111, 0b1111);
log(typeof 0o10, 0o10);
log(typeof 0x12, 0x12);

log(typeof 12345678901234567890, 12345678901234567890);

// big int
log(typeof 12345678901234567890n, 12345678901234567890n);
log(
  typeof 1234567890123456789012345678901234567890123456789012345678901234567890n,
  1234567890123456789012345678901234567890123456789012345678901234567890n
);
```


型 typeof (実行結果)

```
$ deno run ex2.ts
number 1
number 10.1
number 0.000123
number 15
number 8
number 18
number 12345678901234567000
bigint 12345678901234567890n
bigint 1234567890123456789012345678901234567890123456789001234567890n
```


型 typeof function, string, boolean, object

```
// ex3.ts
const log = console.log;

// function
log(typeof log);

// string
log(typeof 'string', 'string');
log(typeof "string", "string");

// boolean
log(typeof false, false);
log(typeof true, true);

// object
const o = { x: 1.1, y: 3, z: 4.4 };
log(typeof o, o);

const a = [1.1, 3, 4.4];
log(typeof a, a);
```


型 typeof (実行結果)

```
$ deno run ex3.ts  
function  
string string  
string string  
boolean false  
boolean true  
object { x: 1.1, y: 3, z: 4.4 }  
object [ 1.1, 3, 4.4 ]
```


型推論

明示的に型を記述しなくていい

javaと同じ **<型>** も使用できる

```
// ex4.ts
const log = console.log;

//
let a = 1;

let b: number;

let c = 3 as number;
let d = <number>4;

b = 2;

log(typeof a, a);
log(typeof b, b);
log(typeof c, c);
log(typeof d, d);
log(typeof log, log);
```

```
$ deno run ex4.ts
number 1
number 2
number 3
number 4
```


JSX

jsxでは **<tag>** を使用できる

```
ReactDOM.render(  
  <h1>Hello, world!</h1>,  
  document.getElementById('root')  
)
```


Lesson2

运算符

演算子

算術演算子	<div><div>+</div><div>-</div></div> <div><div>/</div><div>*</div></div> <div><div>%</div><div>**</div></div>			
等価演算子	<div><div>==</div><div>!=</div><div>===</div><div>!==</div></div>			
関係演算子	<div><div>in</div><div>instanceof</div></div> <div><div><</div><div>></div></div> <div><div><=</div><div>>=</div></div>			
単項演算子	<div><div>delete</div><div>void</div></div> <div><div>typeof</div></div> <div><div>+</div><div>-</div></div> <div><div>~</div><div>!</div></div>			
インクリメント & デクリメント	<div><div>i++</div><div>i--</div><div>++i</div><div>--i</div></div>			
ビット演算子	<div><div>&</div><div> </div><div>^</div></div>			
論理演算子	<div><div>&&</div><div> </div></div>			
代入演算子	<div><div>=</div><div>*=</div><div>/=</div></div> <div><div>%=</div><div>+=</div><div>-=</div></div> <div><div><<=</div><div>>>=</div><div>>>>=</div></div> <div><div>&=</div><div>^=</div><div> =</div></div>			

演算子 2

代入演算子 2

```
const list = [1, 2, 3];  
let [a, b, c, d] = list;  
  
const o = { x: 1, y: 2, z: 3 };  
let { x, z } = o;
```

条件(三項)演算子

```
condition ? ifTrue : ifFalse
```

スプレッド構文

```
...obj
```


演算子 例1

```
// ex1.ts
const log = console.log;

//
log(1 == '1');
log(1 === '1');

log(0 == false);
log(0 === false);

//
log(null == undefined);
log(null === undefined);
```

```
$ deno run ex1.ts
true
false
true
false
true
false
```


演算子 例1 (続き)

拡張子 `.js` も試しましょう

```
$ deno check ex1.ts
Check file:///Users/shingo1551/Documents/github/course/ts1/lesson2/ex1.ts
error: TS2367 [ERROR]: This condition will always return 'false' since the
types 'number' and 'string' have no overlap.
log(1 == '1');
~~~~~
  at file:///Users/shingo1551/Documents/github/course/ts1/lesson2/ex1.ts:5:5

TS2367 [ERROR]: This condition will always return 'false' since the types
'number' and 'string' have no overlap.
log(1 === '1');
~~~~~
  at file:///Users/shingo1551/Documents/github/course/ts1/lesson2/ex1.ts:6:5

TS2367 [ERROR]: This condition will always return 'false' since the types
'number' and 'boolean' have no overlap.
log(0 == false);
~~~~~
  at file:///Users/shingo1551/Documents/github/course/ts1/lesson2/ex1.ts:8:5

TS2367 [ERROR]: This condition will always return 'false' since the types
'number' and 'boolean' have no overlap.
log(0 === false);
~~~~~
  at file:///Users/shingo1551/Documents/github/course/ts1/lesson2/ex1.ts:9:5

Found 4 errors.
```

演算子 例2

いくつか試してみましょう

```
// ex2.ts
const log = console.log;

//
const list = [1, 2, 3];
let [a, b, c, d] = list;
log(a, b, c, d);

[a, b] = [4, 5, 6, 7];
log(a, b);

//
const o = { x: 1, y: 2, z: 3 };
let { x, z } = o;
log(x, z);
```

```
$ deno run ex2.ts
1 2 3 undefined
4 5
1 3
```


演算子 例3

いくつか試してみましょう

```
// ex3.ts
const log = console.log;

//
const m = [1, 2];
const n = [7, 8, 9];
const l = [...m, ...n];
log(l);

//
const o = { a: 1, b: 2 };
const p = { x: 1, y: 2, z: 3 };
const q = { ...o, ...p };
log(q);

//
const a = [...'abcxyz'];
log(a);
```

```
$ deno run ex3.ts
[ 1, 2, 7, 8, 9 ]
{ a: 1, b: 2, x: 1, y: 2, z: 3 }
[ "a", "b", "c", "x", "y", "z" ]
```

Lesson3

予約語、識別子

予約語

Reserved Words	break case catch class const continue debugger default delete do	else enum export extends false finally for function if import	in instanceof new null return super switch this throw true	try typeof var void while with
Strict Mode Reserved Words	as implements interface let package	private protected public static yield		
Contextual Keywords	any boolean constructor declare get	module require number set string	symbol type from of	
その他、定義済	NaN Infinity			

識別子

変数名、関数名など...

大文字と小文字は区別

演算子、予約語など、使用できない

特殊文字で、_ \$ は使用可

0-9 数字で始まるものは不可

```
// ex1.ts
const _a = '_a';
const $a = '$a';

const あいうえお = 'あいうえお';
```


Lesson4

分岐

if...else

if...else

else はなくてもよい

```
// ex1.ts
const a = 10;

if (a > 0)
  console.log('positive');
else
  console.log('NOT positive');
```

```
$ deno run ex1.ts
positive
```

{ }で、複数行の処理を書ける

elseの後にifを書くことができる

```
if (x > 50) {
  ...
  ...
} else if (x > 5) {
  ...
  ...
} else {
  ...
}
```


if...else (続き)

条件、はtrue/false以外の値でもよい

true -> **truthy**

false -> **falsy**

```
// ex2.ts
const a = 10;

if (a)
  console.log('NOT zero');
else
  console.log('zero');
```

```
$ deno run ex2.ts
NOT zero
```


truthy, falsy

falsy

```
false  
0  
-0  
0n  
''  
null  
undefined  
NaN
```

truthy (falsy以外)

```
true  
10  
-2  
1n  
"a"  
'b'  
{ }  
[ ]  
Infinity
```


falsy

!!

```
// ex3.ts
const log = console.log;

//
log(false, !!false);
log(0, !!0);
log(-0, !!-0);
log(0n, !!0n);
log('', !!'');
log(null, !!null);
log(undefined, !!undefined);
log(NaN, !!NaN);
```

```
$ deno run ex3.ts
false false
0 false
-0 false
0n false
false
null false
undefined false
NaN false
```

truthy

```
// ex4.ts
const log = console.log;

//
log(true, !!true);
log(10, !!10);
log(-2, !!-2);
log(1n, !!1n);
log('a', !!'a');
log({}, !!{});
log([], !![]);
log(Infinity, !!Infinity);
```

```
$ deno run ex4.ts
true true
10 true
-2 true
1n true
a true
{} true
[] true
Infinity true
```


switch, case, break

数値や文字列で分岐

default (どの**case**にも一致しない)

breakで途中から抜ける

```
// ex5.ts
const expr = 'Papayas' as string;

switch (expr) {
  case 'Oranges':
    console.log('Oranges are $0.59 a pound. ');
    break;

  case 'Mangoes':
  case 'Papayas':
    console.log('Mangoes and papayas are $2.79 a pound. ');
    break;

  default:
    console.log('Sorry, we are out of ' + expr + '. ');
}
```

```
$ deno run ex5.ts
Mangoes and papayas are $2.79 a pound.
```

Lesson5

繰り返し

while, do...while

```
while (条件) {  
}
```

```
// ex1.ts  
let n = 0;  
  
while (n < 3) {  
    n++;  
}  
  
console.log(n);
```

```
$ deno run ex1.ts  
3
```

```
do {  
  
} while (条件);
```

```
// ex2.ts  
let result = '';  
let i = 0;  
  
do {  
    i = i + 1;  
    result = result + i;  
} while (i < 5);  
  
console.log(result);
```

```
$ deno run ex2.ts  
12345
```

Tips

```
' ' + i  
i.toString()
```

```
+s  
Number.parseFloat(s)  
Number.parseInt(s)
```

```
Boolean(i)
```

```
Boolean(s)
```

```
// ex3.ts  
const log = console.log;  
  
const i = 999;  
log(typeof ' ' + i);  
log(typeof i.toString());  
  
const s = '12345.67';  
log(typeof +s);  
log(typeof Number.parseFloat(s));  
log(typeof Number.parseInt(s));  
  
const b1 = Boolean(i);  
log(typeof b1, b1, !!i);  
  
const b2 = Boolean(s);  
log(typeof b2, b2, !!s);
```

```
$ deno run ex3.ts  
string999  
string  
number  
number  
number  
boolean true true  
boolean true true
```


for

```
for (初期化式; 条件式; 加算式) {  
    文  
}
```

```
// ex4.ts  
const array = ['a', 'b', 'c'];  
  
for (let i = 0; i < array.length; i++)  
    console.log(array[i]);  
  
for (let i = 0; i < 5; i++)  
    console.log(array[i]);
```

```
deno run ex4.ts  
a  
b  
c  
a  
b  
c  
undefined  
undefined
```

for...in, for...of

```
for (variable in object) {  
  文  
}
```

```
// ex5.ts  
const obj = { a: 1, b: 2, c: 3 };  
for (const key in obj)  
  console.log(key);
```

```
$ deno run ex5.ts  
a  
b  
c
```

```
for (variable of iterable) {  
  文  
}
```

```
// ex6.ts  
const array = ['a', 'b', 'c'];  
  
for (const element of array)  
  console.log(element);  
  
for (const element in array)  
  console.log(element);
```

```
$ deno run ex6.ts  
a  
b  
c  
0  
1  
2
```


break, continue

breakの例

forでも使用できる

```
// ex7.ts
let i = 0;
while (i < 6) {
  if (i === 3)
    break;
  i = i + 1;
}
console.log(i);
```

```
$ deno run ex7.ts
3
```

continueの例

whileでも使用できる

```
// ex8.ts
let text = '';
for (let i = 0; i < 10; i++) {
  if (i === 3)
    continue;
  text = text + i;
}
console.log(text);
```

```
deno run ex8.ts
012456789
```


label

labelの例

continueでも使用できる

```
// ex9.ts
loop:
for (let i = 0; i < 3; i++) {
  for (let j = 0; j < 3; j++) {
    if (i === 1 && j === 1)
      break loop;
    console.log('i = ' + i + ', j = ' + j);
  }
}
```

```
$ deno run ex9.ts
i = 0, j = 0
i = 0, j = 1
i = 0, j = 2
i = 1, j = 0
```


Question

実行結果になるようにコードを追加してください

```
const colors = ['red', 'green', 'blue'];
```

ここにコードを追加する

実行結果

注意、**blue**の後に **:** はありません

```
$ deno run colors.ts  
red:green:blue
```


Answer

Answer 1

```
// ans1.ts
const colors = ['red', 'green', 'blue'];

let s = '';
for (let i = 0; i < colors.length; i++) {
  s += colors[i];
  if (i !== 2)
    s += ':';
}
console.log(s);
```

Answer 2

`Array.prototype.join()`

```
// ans2.ts
const colors = ['red', 'green', 'blue'];

console.log(colors.join(':'));
```

TypeScript入門2 で便利な関数を紹介します

Appendix

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>