

# JSの基礎を学習してみよう1



The screenshot shows a browser's developer console with the following output:

```
未成年です。 foundation.js:8
入力値までの数値: foundation.js:12
1 foundation.js:15
2 foundation.js:15
3 foundation.js:15
4 foundation.js:15
5 foundation.js:15
6 foundation.js:15
7 foundation.js:15
8 foundation.js:15
9 foundation.js:15
10 foundation.js:15
1から10までの合計は: 55 foundation.js:26
```

# この教材でできること

- JAVAScript(JS)について理解できる

## どんな教材？

consoleで入出力をし、分岐や繰り返し処理で出力を変更できるようにする。

# 目次

①JSで入出力を試してみよう

②if文を使って出力をしよう

③繰り返し文を使って出力をしよう

# 目次

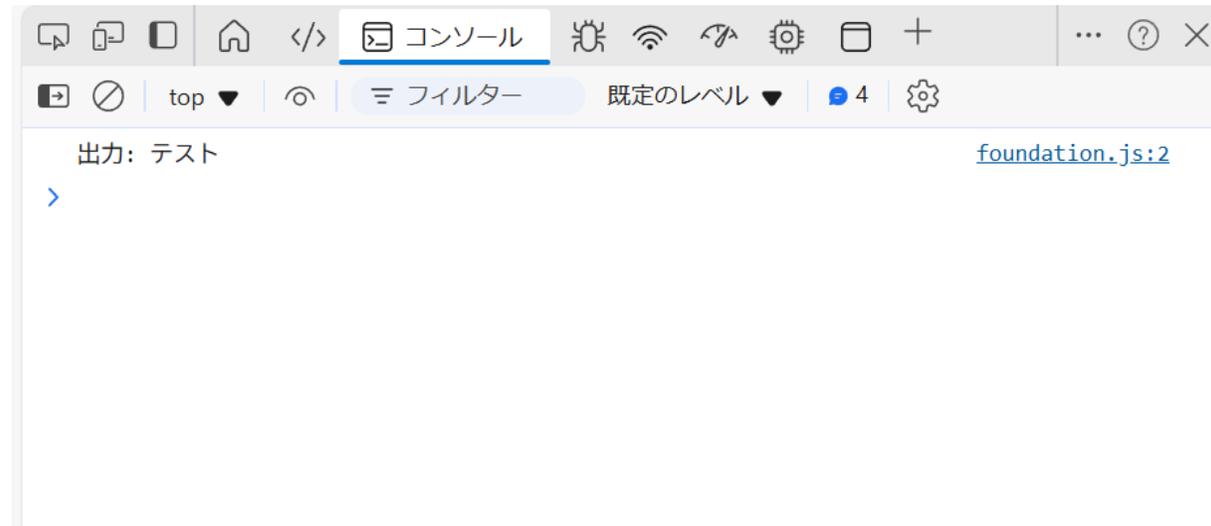
①JSで入出力をしてみよう

②if文を使って出力をしよう

③繰り返し文を使って出力をしよう

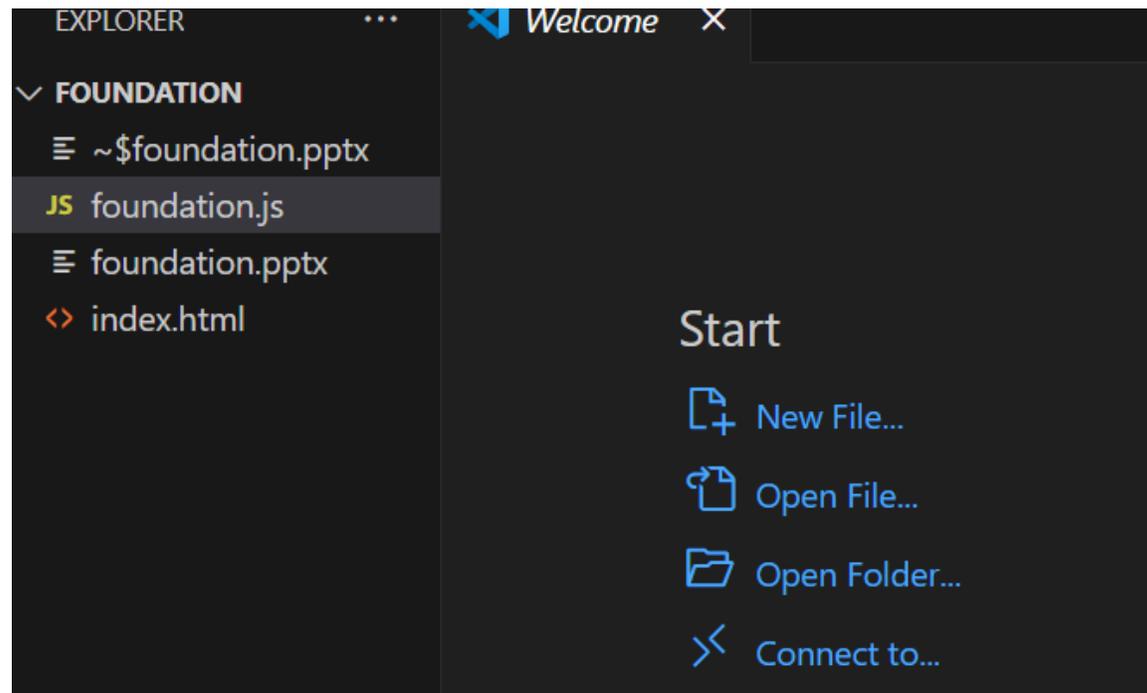
# ①JSで入出力をしてみよう

このページでは入力した値がconsole上で表示されるようにするよ



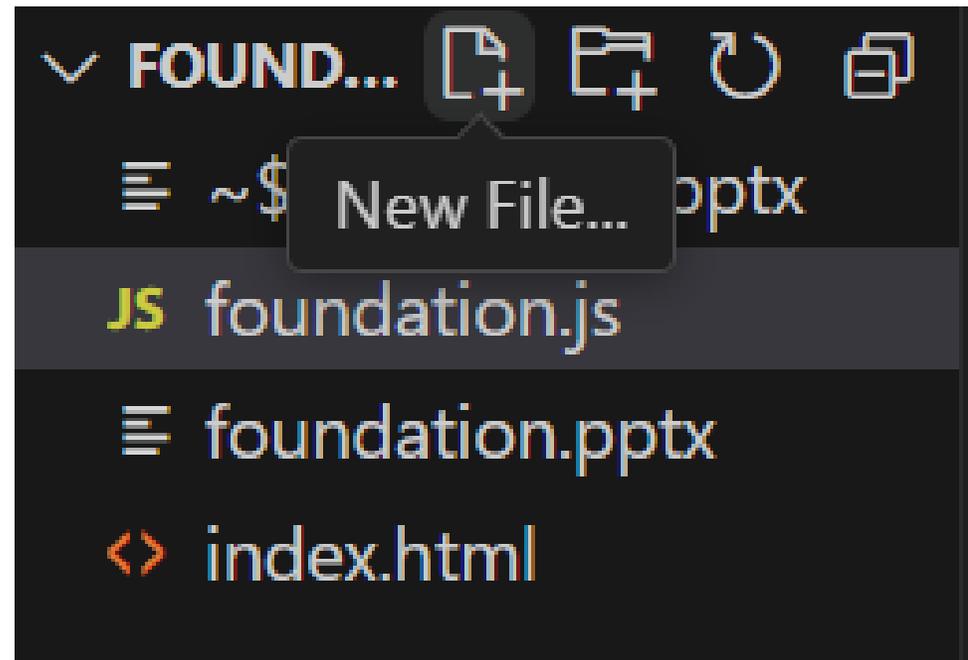
# ①JSで入出力を試してみよう

まずはVscodeを開いてみよう



# ①JSで入出力をしてみよう

htmlとjsのコードを作ってみよう



# 質問

htmlのコードとjsのコードを連携させるようなプログラミングを書いてみよう

※考えてから次のスライドに進んでみよう！

# 答え

htmlファイルに以下の内容を書いたらいいよ

```
<> index.html >  script  
1 <script src="foundation.js"></script>
```

# ①JSで入出力を試してみよう

jsに以下のコードを書いてみよう

```
1  const input = prompt("何か入力してください:"); // 入力を受け取る  
2  console.log("出力: " + input); // 入力を出力
```

# 説明

「const」は変数の宣言をするコマンドだよ

「const input」でinputという名前の変数を宣言しているよ

※変数とは値がどんどん変わっていくような文字や数字のことだよ  
例)体力、レベル、住所

# 説明

promptはダイアログを開くコマンドだよ  
()の中で質問の内容を書いているよ

# 説明

`const input = prompt("何か入力してください:");` ← セミコロンは必ずつける  
日本語では「。」と同じ意味だよ

プログラミングでの「=」は代入という意味になるよ  
promptで何かを入力した値をinput変数に代入する意味になる

# 質問

では算数の「 $3+5=8$ 」の $=$ のように同じ値のような役割をプログラミングで表す場合はどうすればいいかわかるかな？

※考えてから次のスライドに進んでみよう！

# 答え

=を2回書いたらいいよ

例えば「3+5==8」のような使い方をするよ

# 説明

```
console.log("出力: " + input);
```

console.logで文字を出力する意味になるよ  
htmlではpタグのような役割になるよ

「出力:」という文字を表示してinput変数の中の値を表示するよ

# ①JSで入出力をしてみよう

ここまで理解出来たらhtmlを実行してwebページを開いてみよう。写真のような画面が出てきたかな？



# ①JSで入出力を試してみよう

文字化けが出てしまったら、htmlの1行目にmetaタグを書いたら  
文字化けが治るよ

```
<> index.html >  script  
1 <meta charset="UTF-8">  
2 <script src="foundation.js"></script>
```

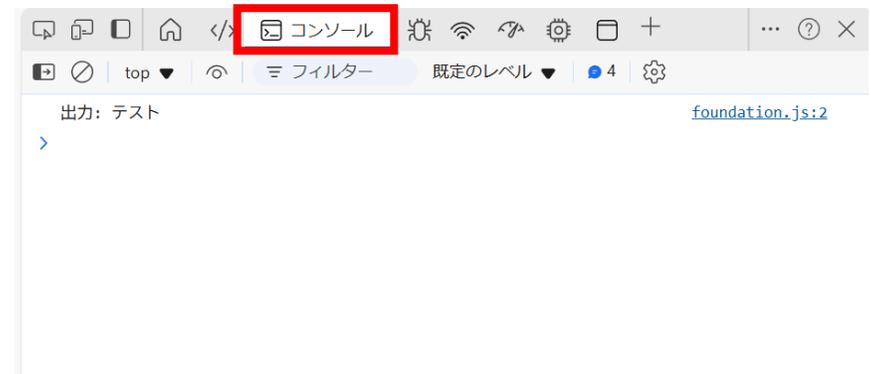
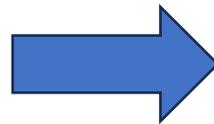
# ①JSで入出力を試してみよう

何か文字を入力してみよう

入力したら右クリックで「開発者ツールで調査しよう」->「コンソール」をクリックするとさっき入力した値が表示されているよ

このページの内容:  
何か入力してください:

OK キャンセル



# 目次

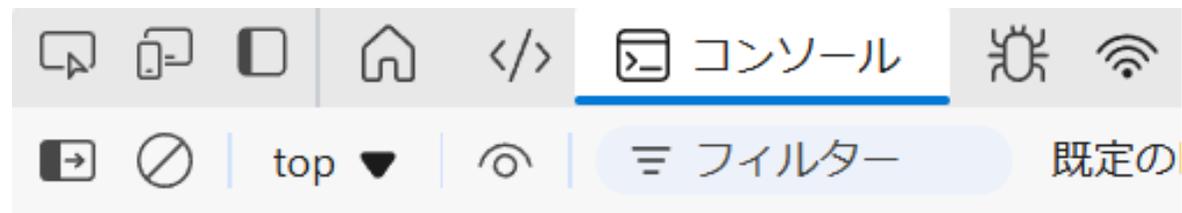
①JSで入出力をしてみよう

②if文を使って出力をしよう

③繰り返し文を使って出力をしよう

## ②if文を使って出力をしよう

このページでは入力した値で出力を変えるようにするよ



成人です。



## ②if文を使って出力をしよう

整数を入力して入力を整数に変換するようにしよう

```
1  const age = prompt("年齢を入力してください:"); // 入力を受け取る
2  const ageNumber = parseInt(age, 10); // 入力を整数に変換
```

# 説明

parseIntでint型の数字にすることができるよ

int型:整数(0、29、-10)

float型:小数(10.5、-10.20)

他にもいろんな型があるから自分で調べてみよう

## ②if文を使って出力をしよう

入力した値が20を超えているかどうかで出力を変えてみよう

```
4  if (ageNumber >= 20) {  
5      console.log("成人です。");  
6  }  
7  else {  
8      console.log("未成年です。");  
9  }
```

# 説明

if文:()の中の条件が満たしていたら{}内を実行する。

※()の中の条件を満たしている場合をtrue、満たしていない場合をfalseという

満たしていない場合でelseifで新しい条件を追加することもできるし、elseで満たしていない場合のコードも書くことができるよ

## ②if文を使って出力をしよう

htmlを実行して数字を入力してみよう

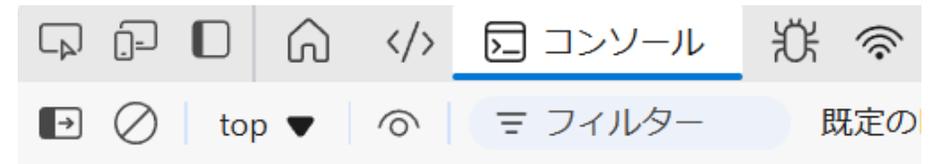
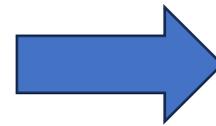
値によって出力が変わるかな？

このページの内容:

年齢を入力してください:

OK

キャンセル



成人です。

>

# 質問

数字以外の文字を入力した場合はどうなるかわかるかな？

※考えてから次のスライドに進んでみよう！

# 答え

試しにテストを入力したら未成年と表示されるよ

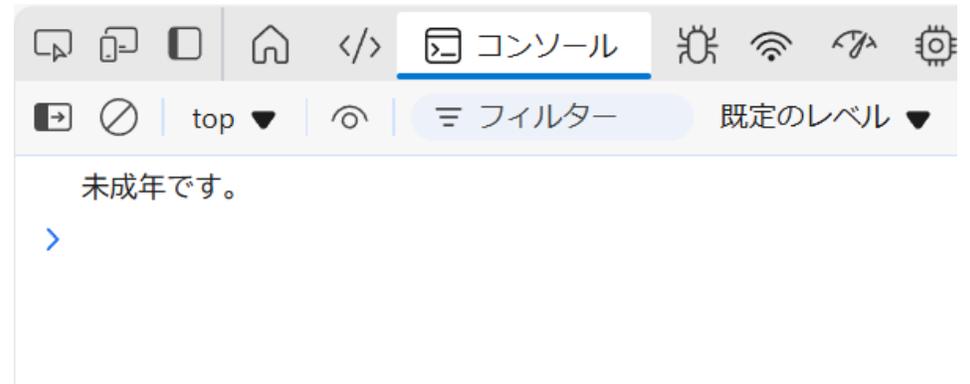
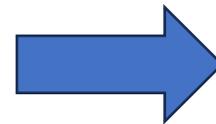
このページの内容:

年齢を入力してください:

テスト

OK

キャンセル



# 答え

これはif文の20以上ではないと判断してすべてelse文の方が実行されることが原因だよ

```
4  if (ageNumber >= 20) {  
5      console.log("成人です。");  
6  }  
7  else {  
8      console.log("未成年です。");  
9  }
```

## ②if文を使って出力をしよう

数字以外の場合は出力を変えてみよう

```
4   if (!isNaN(ageNumber)) { // 有効な数字か確認
5       if (ageNumber >= 20) {
6           console.log("成人です。");
7       } else {
8           console.log("未成年です。");
9       }
10  } else {
11      console.log("有効な数字を入力してください。");
12  }
```

# 説明

isNaN(ageNumber):

ageNumberが数字ではない(NaNはnot a numberの略)場合

「!」をつけると否定の意味になるよ

if(! isNaN(ageNumber)):ageNumberが数字の場合

## ②if文を使って出力をしよう

ここまで出来たらhtmlを実行してみよう

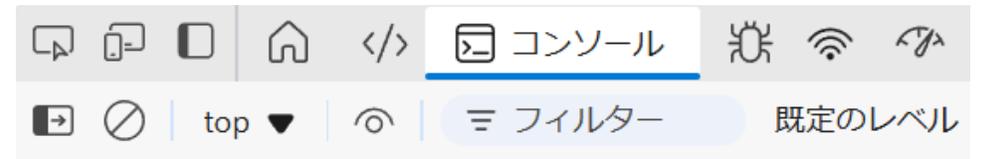
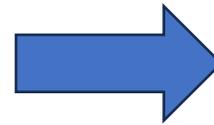
このページの内容:

年齢を入力してください:

テスト

OK

キャンセル



有効な数字を入力してください。

>

# 目次

①JSで入出力をしてみよう

②if文を使って出力をしよう

③繰り返し文を使って出力をしよう

## ③繰り返し文を使って出力をしよう

このページでは入力した値の回数分だけ繰り返し処理をするようにするよ



```
未成年です。 foundation.js:8
入力値までの数値: foundation.js:12
1 foundation.js:15
2 foundation.js:15
3 foundation.js:15
4 foundation.js:15
5 foundation.js:15
6 foundation.js:15
7 foundation.js:15
8 foundation.js:15
9 foundation.js:15
10 foundation.js:15
1から10までの合計は: 55 foundation.js:26
```

## ③繰り返し文を使って出力をしよう

繰り返しの方法は3種類あるよ

do文:後判定(実行した後に繰り返すか判断する)

while文:前判定(判断して繰り返しの処理を実行する)

for文:前判定(判断して繰り返しの処理を実行する)

## ③繰り返し文を使って出力をしよう

まずはdo文を使って処理をするよ

```
4  if (!isNaN(ageNumber)) { // 有効な数字か確認
5      if (ageNumber >= 20) {
6          console.log("成人です。");
7      } else {
8          console.log("未成年です。");
9      }
10
11     // 入力値を1から順に表示 (do...while)
12     console.log("入力値までの数値:");
13     let i = 1;
14     do {
15         console.log(i);
16         i++;
17     } while (i <= ageNumber);
18
```

# 説明

```
do{  
    コード  
}while(繰り返し条件);
```

do文の中を実行して終わったらwhileの()で繰り返すかの判断を行う

# 説明

`i++` : `i`の値を1増やす

`i--` : `i`の値を1減らす

例)

```
level = 5
```

```
if(経験値がマックスになったら){
```

```
    経験値変数 = 0
```

```
    level++
```

```
}
```

# ③繰り返し文を使って出力をしよう

htmlを実行して数字を入力してみよう

成人です。	<a href="#">foundation.js:6</a>
入力値までの数値:	<a href="#">foundation.js:12</a>
1	<a href="#">foundation.js:15</a>
2	<a href="#">foundation.js:15</a>
3	<a href="#">foundation.js:15</a>
4	<a href="#">foundation.js:15</a>
5	<a href="#">foundation.js:15</a>
6	<a href="#">foundation.js:15</a>
7	<a href="#">foundation.js:15</a>
8	<a href="#">foundation.js:15</a>
9	<a href="#">foundation.js:15</a>
10	<a href="#">foundation.js:15</a>
11	<a href="#">foundation.js:15</a>
12	<a href="#">foundation.js:15</a>
13	<a href="#">foundation.js:15</a>
14	<a href="#">foundation.js:15</a>
15	<a href="#">foundation.js:15</a>
16	<a href="#">foundation.js:15</a>
17	<a href="#">foundation.js:15</a>
18	<a href="#">foundation.js:15</a>
19	<a href="#">foundation.js:15</a>
20	<a href="#">foundation.js:15</a>

## ③繰り返し文を使って出力をしよう

次にwhile文を使って1から入力した数字までの合計を表示させよう

```
18
19 // 1から入力した数字までの合計を計算 (while)
20 let sum = 0;
21 let j = 1;
22 while (j <= ageNumber) {
23     sum += j;
24     j++;
25 }
26 console.log(`1から${ageNumber}までの合計は: ${sum}`);
```

# 説明

```
while(繰り返し条件){  
    コード  
}
```

繰り返しの条件を満たしているか確認してコードの処理を行う

# 説明

sum += j : sumの値にjの数だけ加算する

「sum = sum + j」を省略したら「sum += j」となる  
同じ変数

例)

```
sum = 10
```

```
sum += 5    //sumは15となる
```

```
age = 12
```

```
age -= 7    //ageは5となる
```

# ③繰り返し文を使って出力をしよう

htmlを実行して結果を見てみよう

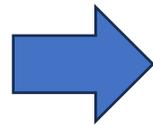


```
未成年です。 foundation.js:8
入力値までの数値: foundation.js:12
1 foundation.js:15
2 foundation.js:15
3 foundation.js:15
4 foundation.js:15
5 foundation.js:15
6 foundation.js:15
7 foundation.js:15
8 foundation.js:15
9 foundation.js:15
10 foundation.js:15
1から10までの合計は: 55 foundation.js:26
```

## ③繰り返し文を使って出力をしよう

while文をfor文に変更してみよう

```
18
19 // 1から入力した数字までの合計を計算 (while)
20 let sum = 0;
21 let j = 1;
22 while (j <= ageNumber) {
23     sum += j;
24     j++;
25 }
26 console.log(`1から${ageNumber}までの合計は: ${sum}`);
```



```
18
19 // 1から入力した数字までの合計を計算 (for)
20 let sum = 0;
21 for (let j = 1; j <= ageNumber; j++) {
22     sum += j;
23 }
24 console.log(`1から${ageNumber}までの合計は: ${sum}`);
25
```

# 説明

```
for(変数の宣言 ; 繰り返しの条件を決める ; 変数の値を変える){  
    コード  
}
```

例)

```
len j = 1;  
while (j <= ageNumber) {  
    sum += j;  
    j++;  
}  
for (let j = 1; j <= ageNumber; j++) {  
    sum += j;  
}
```

## ③繰り返し文を使って出力をしよう

htmlを実行してwhile文と同じような結果になるか見てみよう



```
未成年です。 foundation.js:8
入力値までの数値: foundation.js:12
1 foundation.js:15
2 foundation.js:15
3 foundation.js:15
4 foundation.js:15
5 foundation.js:15
6 foundation.js:15
7 foundation.js:15
8 foundation.js:15
9 foundation.js:15
10 foundation.js:15
1から10までの合計は: 55 foundation.js:26
>
```

# お疲れさまでした

テキストは終了です。  
あとは自分なりにアレンジを付け加えていこう！