




Web Developer Bootcamp

Function (関数)

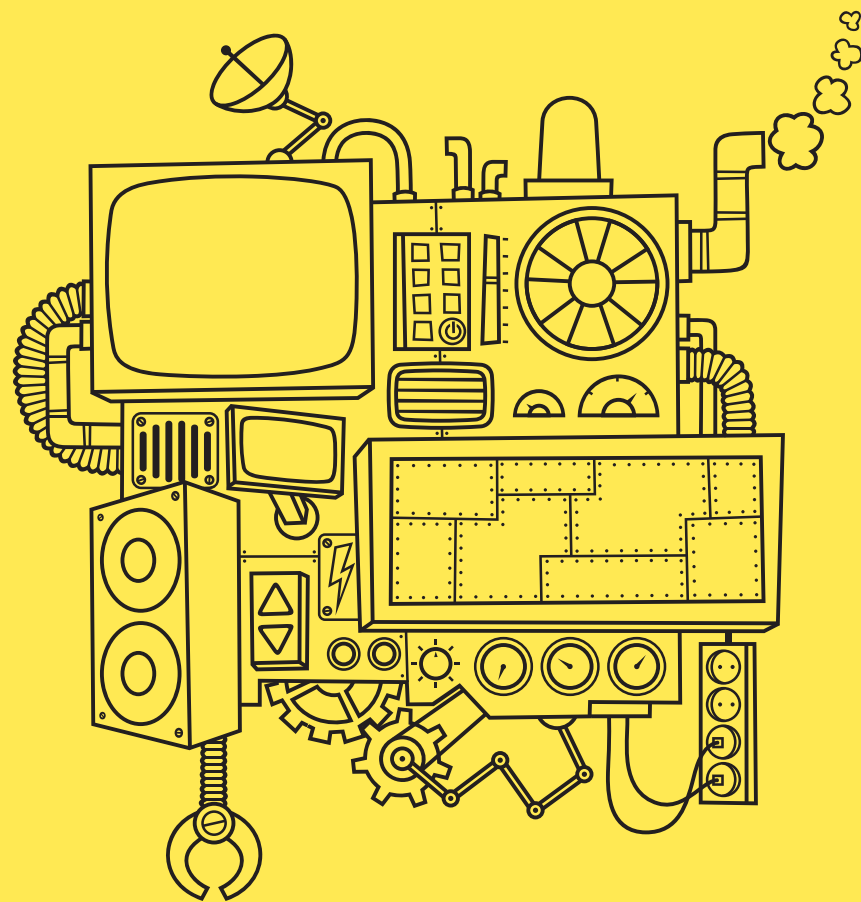
最後の「大きな」トピック



関数

再利用可能な処理

- 関数を使うことでコードの再利用が可能になる
- まとまったコードを定義して、後で実行できる
- あらゆる場面で使う



ポイントは2つ

定義



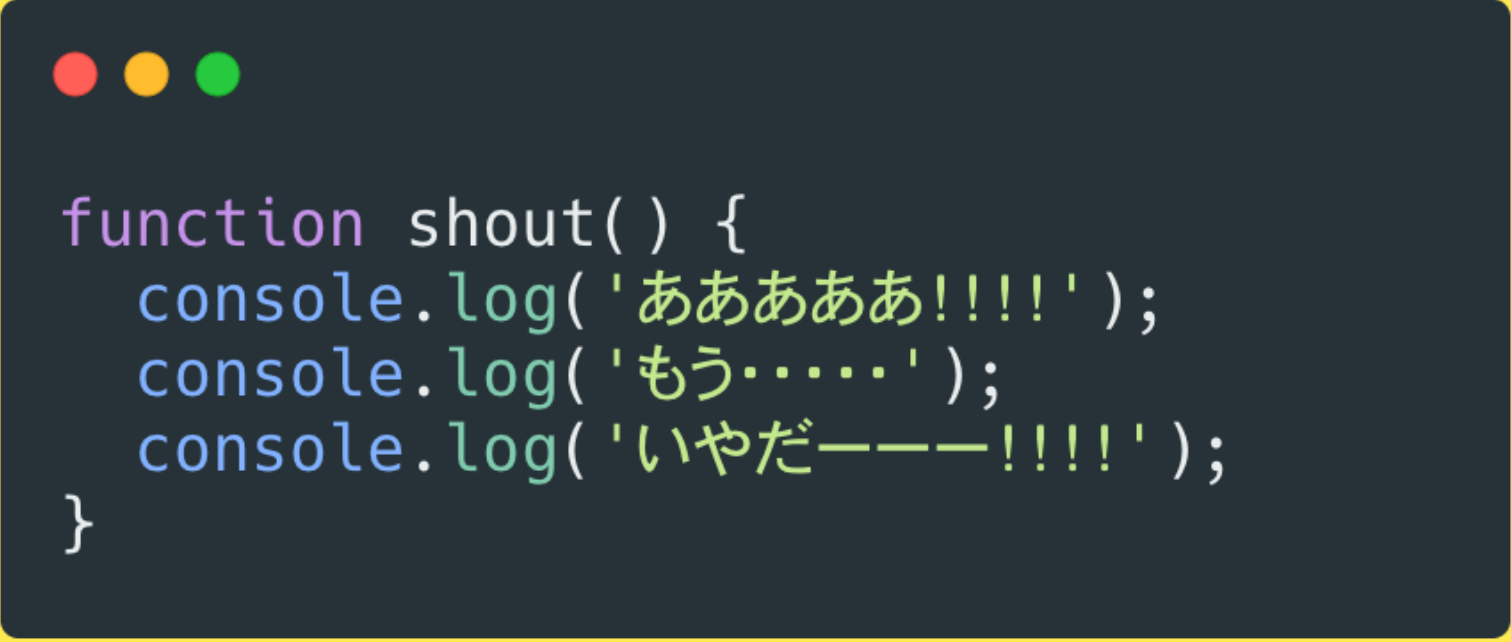
実行



定義

```
function funcName(){  
    //何らかの処理  
}
```

定義



```
function shout() {  
  console.log( 'あああああ!!!!' );  
  console.log( 'もう……' );  
  console.log( 'いやだ———!!!!' );  
}
```

実行

```
funcName(); //実行
```

```
funcName(); //また実行
```

実行

```
shout();  
// あああああ!!!!  
// もう……  
// いやだ———!!!!
```

```
shout();  
// あああああ!!!!  
// もう……  
// いやだ———!!!!
```





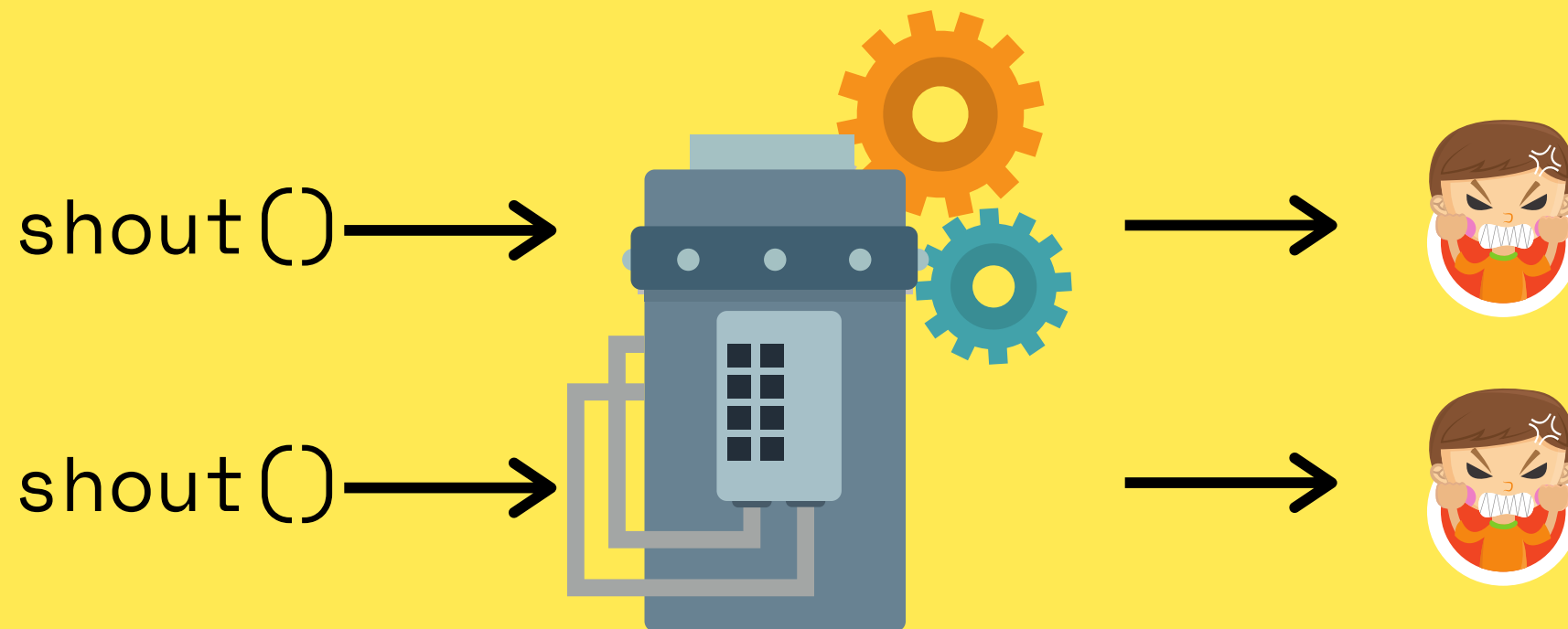
引数

(ARGUMENTS)

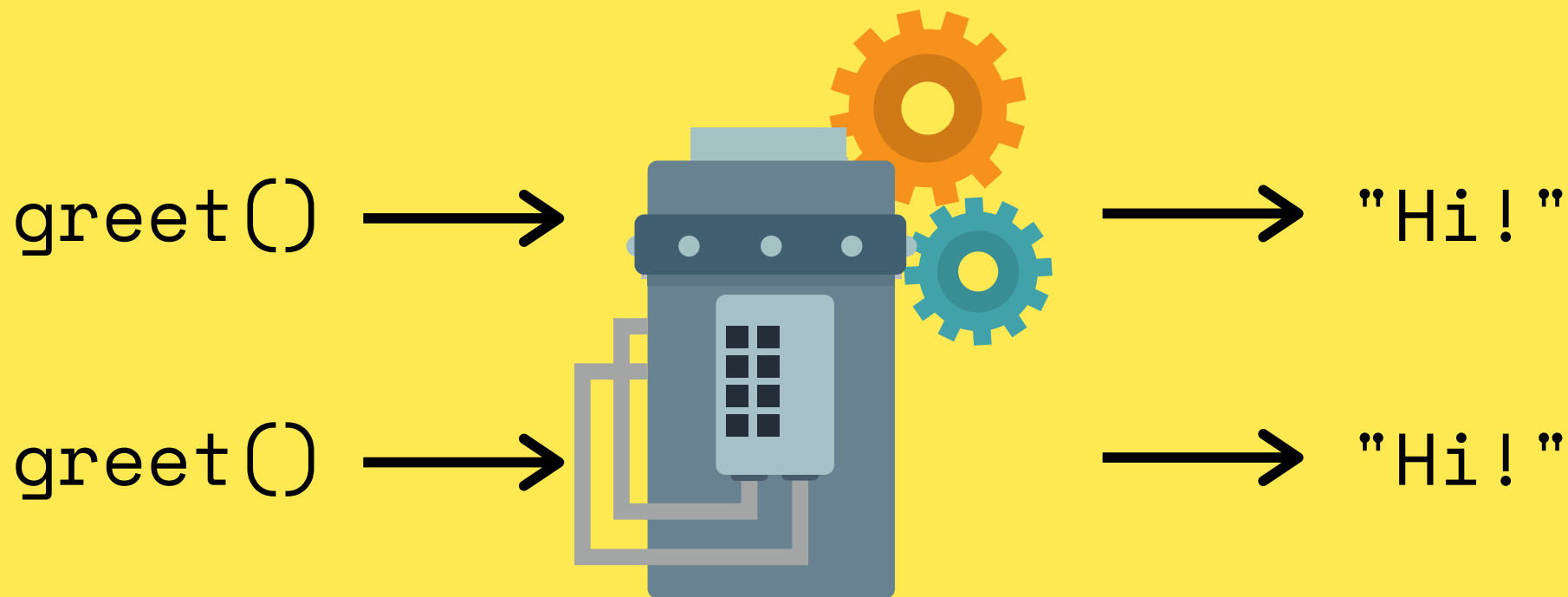
入力値

今のところ作った関数は
入力値を受け取らない。
何度実行しても同じ結果になる。

入力値なし



入力値なし

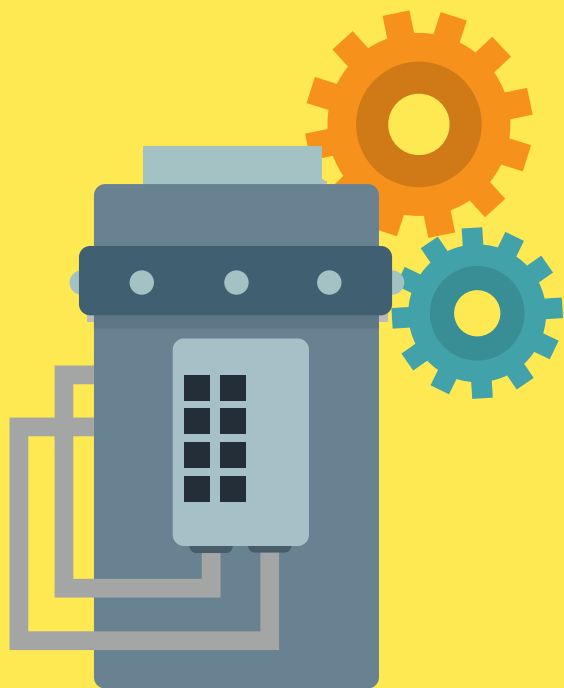


引数

入力値を受けつける関数も定義できる。
この入力値のことを引数（ひきすう）と呼ぶ

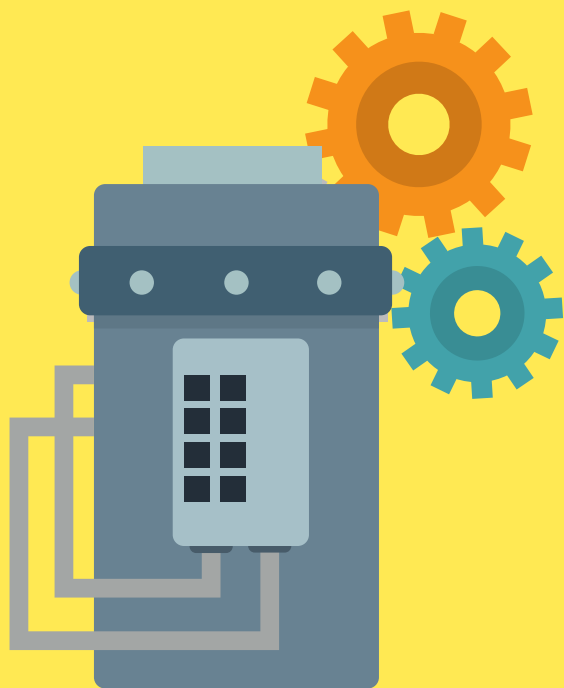
引数

`greet('Tim')`



`"Hi Tim!"`

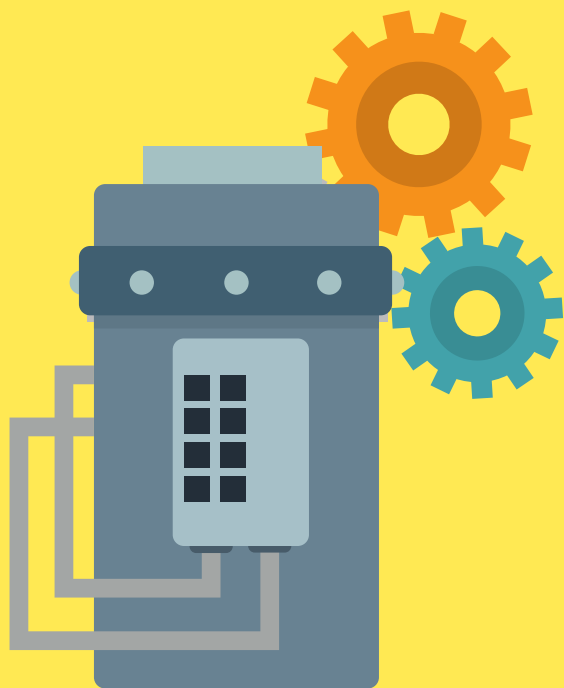
`greet('Aya')`



`"Hi Aya!"`

引数

`avg(20, 25)`



22.5

`avg(3, 2, 5, 6)`



4

今までも使っています


引数なし

```
// 引数なし  
'hello'.toUpperCase();
```

引数あり

```
// 引数が違うと  
'hello'.indexOf('h'); //0  
// 結果も違う  
'hello'.indexOf('o'); //4
```

GREET



```
function greet(person) {  
  console.log(`Hi, ${person}!`);  
}
```



```
greet('Aya');
```



"Hi, Aya!"



```
greet('Ned');
```



"Hi, Ned!"

複数の引数

```
function findLargest(x, y) {  
  if (x > y) {  
    console.log(`${x} is larger!`);  
  }  
  else if (x < y) {  
    console.log(`${y} is larger!`);  
  }  
  else {  
    console.log(`${x} and ${y} are equal!`);  
  }  
}
```

```
findLargest(-2,77)
```

"77 is
larger!"

```
findLargest(33,33);
```

"33 and 33
are equal"

RETURN

組み込みメソッドは実行すると値が返ってくる（returnされる）。
この値を保持しておける：



```
const yell = "I will end you".toUpperCase();
```

```
yell; //"I WILL END YOU"
```

```
const idx = ['a', 'b', 'c'].indexOf('c');
```

```
idx; //2
```

RETURNなし!

今まで作った関数は値を出力するけど、
returnはしていない



```
function add(x, y) {  
  console.log(x + y);  
}  
  
const sum = add(10, 16);  
sum; //undefined
```



はじめてのRETURN！

これで変数に保持できる！



```
function add(x, y) {  
  return x + y; //RETURN!  
}  
  
const sum = add(10, 16);  
sum; //26  
  
const answer = add(100, 200);  
answer; //300
```



RETURN

returnで関数をその時点で終わらせ、
関数からどんな値を返すかも指定できる