2023 年 10 月 2 日（月）

『おもしろ電子工作実験室』

●JavaScript の文法とデータ型

基本

JavaScript は**大文字**と**小文字**を区別し、**Unicode** 文字セットを使用する。
Unicode が使用可能なので、例えば変数名に日本語も使用可能。

```
const 名前 = "与論 太郎";
```

JavaScript では、命令は**文**（statement）と呼び、**セミコロン（;）**によって区切られる。
文が単独の行で書かれている場合は文の後のセミコロンは省略可能。行の中に複数の文が必要な場合は、セミコロンで区切る必要がある。

```
led.on()

led.off(); count += 1;
```

文が単独なので ； は省略可能

文が複数の場合は ； で区切る

※文が単独の場合でも、バグの原因になる可能性があるので セミコロン を記述した方が良い。

コメント

コメントとは、プログラムの処理に影響を与えない文章のことで、プログラム開発の覚書を記述し、後で見直したときなどにソースコードが分かりやすいように記述する文章のこと。
JavaScript では、1 行コメントとブロックコメントの 2 つの記述方法がある。

```
// 行の終わりまでコメント

/*
  ブロックコメント
*/
```

// 以降 行の終わりまでがコメント

/* から */ までがコメント

変数の宣言

var	古い宣言方法。グローバル変数の宣言で使用する。
let	ブロックスコープのローカル変数を宣言。
const	ブロックスコープの読取専用のローカル変数を宣言。定数と呼ぶ場合もある。

※ブロックスコープとは if 文や for 文などの処理で { から } までの範囲内でのみ有効なことを意味する。

変数

変数とはプログラム中で使用するデータを一時的に保存する領域のこと。
変数名は、プログラムで値を表す印的な名前として使用する。変数の名前は、識別子と呼び一定のルールに従う必要がある。

JavaScript の識別子は、A ～ Z（大文字)、a ～ z（小文字)、アンダースコア（_)、ドル記号($)、Unicode（日本語など）から始める必要がる。2 文字目以降は、それ以外に 0 ～ 9（数字）も使用できる。

正しい名前の例：bango、_error404、$this_is_a_pen、年齢
正しくない名前の例：55ban （数字から始まっている）

※識別子は大文字と小文字を区別するので､例えば bango と Bango は別識別子として認識。

変数のスコープ

スコープとは、宣言した変数の有効範囲のこと。
下記のプログラムを Paiza（https://paiza.io/）で動作確認を行い、 var と let のスコープの違いを確認してください。

```
// var で宣言した変数のスコープ
for (var i = 0; i < 5; i++) {
    console.log(i);
}
console.log(i);
```

上記 var を let に変更すると？

```
// let で宣言した変数のスコープ
for (let i = 0; i < 5; i++) {
```

let の宣言を for 文の前で宣言すると？

```
let i = 0;
for (; i < 5; i++) {
    console.log(i);
}
console.log(i);
```

let の宣言を for 文の前と for 文の中で同じ変数名を宣言すると？

```
let i = “for 文の外”;
for (let i = 0; i < 5; i++) {
    console.log(i);
}
console.log(i);
```

const も確認！

```
const num = 100;

num = 200;
```

データ型

JavaScript でよく使用するデータ型は以下の通り。

Boolean （論理値）	true または false
null	null 値を意味する特殊なキーワード
undefined （未定義）	値が未定義
Number （数値）	整数または浮動小数点数。例えば 42 や 3.14159 など。
String （文字列）	テキストの値を表す連続した文字。"お名前" など。
Object （オブジェクト）	オブジェクト

※これ以外にも BigInt や Symbol がある。

データ型の使いまわし

JavaScript は他の言語（例えば C 言語や Basic）のような明示的に型宣言を行わない。
プログラム実行時に自動的に（勝手に）型変換が行われる。

下記のように記述してもエラーにならない

```
let age = 46;

age = “与論町茶花”;
```

文字列から数値への変換

文字列が数値を表している場合、文字列から数値への変換は下記のメソッドを使用する。

parseInt()　　整数に変換

parseFloat()　　浮動小数点に変換

動作確認を行ってください。

```
let input = “52”;
let age = parseInt(input);
console.log(age);

let input = “46.83”;
let sensor = parseFloat(input);
console.log(sensor);
```

下記の場合はどうかるか？

```
// 0から始まる数値文字列は？
let input = “0052”;

// 空白が入っていたら？
let input = “　　52　　”;


// 数値以外の文字が入っていたら？
let input = “52abc”;
let input = “xyz52abc”;
```

メソッドを使用しないとどうかるか？

```
// 0から始まる数値文字列は？
console.log(let input = “0052”;

// 空白が入っていたら？
let input = “　　52　　”;


// 数値以外の文字が入っていたら？
let input = “52abc”;
let input = “xyz52abc”;
```

文字列の演算

下記がどうなるか確認してください。

```
// 引き算
console.log(12 - 2);
console.log(“12” - 2);
console.log(12 - “2”);
console.log(“12” - “2”);

// 掛け算
console.log(12 * 2);
console.log(“12” * 2);
console.log(12 * “2”);
console.log(“12” * “2”);

// 割り算
console.log(12 / 2);
console.log(“12” / 2);
console.log(12 / “2”);
console.log(“12” / “2”);

// 足し算
console.log(12 + 2);
console.log(“12” + 2);
console.log(12 + “2”);
console.log(“12” + “2”);
```

イコール（＝）演算子

プログラミングで使用するイコール（＝）は、「等しい」という意味ではなく、「**代入**」の意味になります。変数へ値を**代入**するときに使用する。

例：

```
let age = 48;                 // 変数 age に 数値 48 を代入する
let name = “山田太郎”;        // 変数 name に 文字列 “山田太郎”を代入する
let age = 2023 – 1975;        // 右辺の式 2023-1975 を計算し、結果を変数 age に代入する
let str = “年齢は” + 48;      // 右辺の式の数値 48 を文字列に変換し、文字列”年齢は”と
                              // 変換した文字列 “48”を結合し、変数 str に代入する
```

※「==」や「===」は比較演算子、「=」は代入演算子で別の意味
　比較演算子は別の機会に説明します。