関数 function

関数の作り方

関数とは記述をグループ化して、何度でも繰り返し使えるようにすること。

自分で関数名をつける

```
function 関数名(引数){
    // ここに処理内容を記述する
}
```

関数の実行

関数を実行するには命名した「関数名」に`()`をつけると、関数内の文章が実行される。

```
関数名();
```

関数の定義と実行

自分で関数の名前を命名する。半角英数であれば特に命名のルールは無いが、慣習として命令の意味が伝わる単語繋げると良い。単語区切りは２単語目以降を大文字にすると読みやすくなる。

```
// 関数の定義
function changeStyle(){
    document.querySelector('#ttl').style.color = '#FF0000';
}

// 関数の実行
changeStyle();
```

引数

引数とは

引数とは関数内で使用する変数。実行時に関数の中へ値を受け渡すことで、関数内処理ができる。

```
// 関数の定義
function changeStyle(aColor){
    document.querySelector('#ttl').style.color = aColor;
}

// 関数の実行
changeStyle('#00FF00');
```

引数は`,`カンマ区切りで複数受け渡すことができる。

return

関数内で `return` を使うと処理された値を返す（戻す）ことができる。

```
// 関数の定義
function calcNum(aNum1, aNum2){
    var result = aNum1 + aNum2;
    return result;
}

// 関数の実行
var resultNum = calcNum(2,3);

// 実行結果の確認
alert(resultNum); // 結果は5
```

グローバル変数とローカル変数

変数は宣言された場所によって **スコープ（参照できる範囲）** がある。

グローバル変数

関数`{}`の外側で宣言された変数は**グローバル変数**と呼ばれ関数内外で参照することができる。

```
// グローバル変数
var colorValue = '#FF0000';

//  関数の定義
function changeStyle( ){
  document.querySelector('#ttl').style.color = colorValue;
}

//関数の実行
changeStyle();
```

ローカル変数

関数`{}`の内側で宣言された変数は**ローカル変数**と呼ばれ関数内でしか参照することができない。

```
function changeColor( ){

  // ローカル変数
  var colorValue = '#FF0000';
  document.querySelector('#ttl').style.color = colorValue;
}

alert(colorValue); // エラー 変数"colorValue"は見つからない
```

グローバル変数とローカル変数の違い

- グローバル変数はどこからでも参照できるがメモリに残り続けるので負荷が高い。
- ローカル変数は参照範囲が絞られる為、一時的にメモリを消費して実行が終わると破棄されるので負荷が低い。