Параметры функций

Нам надо сложить неограниченное кол-во чисел. В этом может пригодиться остаточные параметры:

function sum (a, ...b) {

let res = a;

for (let num of b) {

res += num;

}

return res;

}

let r = sum(1, 1, 3, 5, 7);

console.log(r);

Но допустим нам нужно, сосчитать массив. Для таких случаев нужно использовать *оператор расширения*:

r = sum(...[1, 4, 6, 8, 9, 12]);

console.log(r);

Хотя оператор используют еще много где. Например, можно преобразовать любой итерируемый объект в массив:

let chars = [...'Hallo, World'];

console.log(chars);

Другое часто используемое – это значение по-умолчанию, т.е. изначально у каждого параметра undefined, но можно указать значение, которое можно не указывать.

function append (values, array=[]) {

for (let i of values) {

array.push (i);

}

return array;

}

let r = append ('Hello');

console.log(r);

let arr = [1,3, 5];

r = append ('Hello', arr);

console.log(r);

Так как функции – это те же объекты, то есть их можно передать в функцию. Можно без трудностей реализовать *обвертку*, например хеширование:

function decorate(func, hash =

(param) => {

return [...param].join(',');

}) {

let cache = new Map();

return function () {

let key = hash(arguments);

if (cache.has(key)) {

return cache.get(key);

}

let result = func(...arguments);

cache.set (key, result);

return result;

}

}

function calc(n) {

return (n != 1) ? n \* calc(n - 1) : 1;

}

f = decorate (calc);

console.log (f (5));

console.log (f (5));

console.log (f (5));

console.log (f (5));

console.log (f (5));

Начнем разбирать. Первое, что мы передаем функцию, которую надо кэшировать. Но у функции может быть несколько параметров, а хэш-таблицы только ключ-значение.

Как вариант из множества – это кэшировать аргументы. У функции hash() уже есть значение по-умолчанию, которое подходит для простых объектов.

Далее создается сама таблица, которая будет содержать заготовленные ответы. Создается сама функция, которая обвертка над func. Сначала она ищет значение, а иначе считает и добавляет ответ с ключом в cache.

В этой функции используется механизм замыкания. А для того чтоб не париться с аргументами для обвертки использовался псевдомассив arguments, который содержит все неявно переданные аргументы.

Этот пример очень содержательный и притом крайне практичный.