Object.keys, values, entries

Давайте отойдём от отдельных структур данных и поговорим об их переборе вообще.

В предыдущей главе мы видели методы `map.keys()`, `map.values()`, `map.entries()`.

Это универсальные методы, и существует общее соглашение использовать их для структур данных. Если бы мы делали собственную структуру данных, нам также следовало бы их реализовать.

Методы поддерживаются для структур:

- `Map`
- `Set`
- `Array`

Простые объекты также можно перебирать похожими методами, но синтаксис немного отличается.

Object.keys, values, entries

Для простых объектов доступны следующие методы:

- Object.keys(obj) – возвращает массив ключей.
- Object.values(obj) – возвращает массив значений.
- Object.entries(obj) – возвращает массив пар `[ключ, значение]`.

Обратите внимание на различия (по сравнению с `map`, например):

	Map	Object
Синтаксис вызова	`map.keys()`	`Object.keys(obj)`, не `obj.keys()`
Возвращает	перебираемый объект	«реальный» массив

Первое отличие в том, что мы должны вызвать `Object.keys(obj)`, а не `obj.keys()`.

Почему так? Основная причина – гибкость. Помните, что объекты являются основой всех сложных структур в JavaScript. У нас может быть объект `data`, который реализует свой собственный метод `data.values()`. И мы всё ещё можем применять к нему стандартный метод `Object.values(data)`.

Второе отличие в том, что методы вида `Object.*` возвращают «реальные» массивы, а не просто итерируемые объекты. Это в основном по историческим причинам.

Например:

```
let user = {
  name: "John",
  age: 30
};
```

- `Object.keys(user) = ["name", "age"]`
- `Object.values(user) = ["John", 30]`
- `Object.entries(user) = [ ["name","John"], ["age",30] ]`

Раздел
Типы данных

Навигация по уроку
Object.keys, values, entries
Трансформации объекта

Задачи (2)
Комментарии

Поделиться

Редактировать на GitHub

Вот пример использования `Object.values` для перебора значений свойств в цикле:

```
let user = {
  name: "John",
  age: 30
};

// перебор значений
for (let value of Object.values(user)) {
  alert(value); // John, затем 30
}
```

Object.keys/values/entries игнорируют символьные свойства

Так же, как и цикл `for..in`, эти методы игнорируют свойства, использующие `Symbol(...)` в качестве ключей.

Обычно это удобно. Но если требуется учитывать и символьные ключи, то для этого существует отдельный метод Object.getOwnPropertySymbols, возвращающий массив только символьных ключей. Также, существует метод Reflect.ownKeys(obj), который возвращает *все* ключи.

Трансформации объекта

У объектов нет множества методов, которые есть в массивах, например `map`, `filter` и других.

Если мы хотели бы их применить, то можно использовать `Object.entries` с последующим вызовом `Object.fromEntries`:

1. Вызов `Object.entries(obj)` возвращает массив пар ключ/значение для `obj`.
2. На нём вызываем методы массива, например, `map`.
3. Используем `Object.fromEntries(array)` на результате, чтобы преобразовать его обратно в объект.

Например, у нас есть объект с ценами, и мы хотели бы их удвоить:

```
let prices = {
  banana: 1,
  orange: 2,
  meat: 4,
};

let doublePrices = Object.fromEntries(
  // преобразовать в массив, затем map, затем fromEntri
  Object.entries(prices).map(([key, value]) => [key, va
);

alert(doublePrices.meat); // 8
```

Это может выглядеть сложным на первый взгляд, но становится лёгким для понимания после нескольких раз использования.

Можно делать и более сложные «однострочные» преобразования таким путём. Важно только сохранять баланс, чтобы код при этом был достаточно простым для понимания.

Задачи

Сумма свойств объекта

важность: 5

Есть объект `salaries` с произвольным количеством свойств, содержащих заработные платы.

Напишите функцию `sumSalaries(salaries)`, которая возвращает сумму всех зарплат с помощью метода `Object.values` и цикла `for..of`.

Если объект `salaries` пуст, то результат должен быть `0`.

Например:

```
let salaries = {
  "John": 100,
  "Pete": 300,
  "Mary": 250
};

alert( sumSalaries(salaries) ); // 650
```

Открыть песочницу с тестами для задачи.

решение

Подсчёт количества свойств объекта

важность: 5

Напишите функцию `count(obj)`, которая возвращает количество свойств объекта:

```
let user = {
  name: 'John',
  age: 30
};

alert( count(user) ); // 2
```

Постарайтесь сделать код как можно короче.

P.S. Игнорируйте символьные свойства, подсчитывайте только «обычные».

Открыть песочницу с тестами для задачи.

решение

Комментарии

перед тем как писать...