Map в JavaScript

`Map` - это структура данных в JavaScript, представляющая собой коллекцию ключ-значение, где ключи могут быть любого типа, и каждому ключу соответствует определенное значение. Основное отличие `Map` от объекта (`Object`) заключается в том, что `Map` позволяет использовать любой тип данных в качестве ключа, включая объекты, функции и другие примитивы. В объектах же ключами могут быть только строки и символы.

Основные характеристики Map:

1. **Уникальность ключей**: Каждый ключ в `Map` уникален, и в отличие от объектов, не может быть дублирующихся ключей.
2. **Порядок элементов**: Элементы в `Map` хранятся в порядке их добавления, что делает их упорядоченными коллекциями.
3. **Любой тип ключа**: В `Map` ключами могут быть не только строки, но и любые другие типы данных, включая объекты, функции и примитивы.

Основные методы Map:

- `set(key, value)`: Добавляет пару ключ-значение в `Map`.
- `get(key)`: Возвращает значение, связанное с указанным ключом.
- `has(key)`: Проверяет наличие ключа в `Map`.
- `delete(key)`: Удаляет пару ключ-значение по ключу.
- `size`: Возвращает количество элементов в `Map`.
- `clear()`: Удаляет все элементы из `Map`.
- `keys()`: Возвращает итератор, позволяющий перебирать ключи.
- `values()`: Возвращает итератор, позволяющий перебирать значения.
- `entries()`: Возвращает итератор, позволяющий перебирать пары ключ-значение.
- `forEach(callback)`: Позволяет итерировать по `Map`, вызывая функцию обратного вызова для каждой пары ключ-значение.

`Map` представляет собой мощную структуру данных, которая может быть использована для организации и хранения данных с произвольными ключами и значениями.

Пример использования Map:

```
const myMap = new Map();

myMap.set("name", "Alice");
myMap.set("age", 30);
myMap.set({ key: "customObject" }, "This is a custom object");

console.log(myMap.get("name")); // "Alice"
console.log(myMap.has("age")); // true

myMap.delete("age");
console.log(myMap.has("age")); // false

console.log(myMap.size); // 2

myMap.forEach((value, key) => {
  console.log(`${key}: ${value}`);
});
// "name: Alice"
// "key: This is a custom object"
```

В этом примере создается `Map` с несколькими парами ключ-значение, и затем используются методы `set()`, `get()`, `has()`, `delete()`, `size` и `forEach()` для работы с данными в `Map`.

```
const employeeMap = new Map();

// Добавление сотрудников
employeeMap.set('John', { age: 30, department: 'HR' });
employeeMap.set('Alice', { age: 28, department: 'Finance' });

// Получение информации о сотруднике
const johnInfo = employeeMap.get('John');
console.log(johnInfo);

// Проверка наличия сотрудника
if (employeeMap.has('Alice')) {
  console.log('Alice работает в компании.');
}

// Итерация по Map
for (const [name, info] of employeeMap) {
  console.log(`${name}: Возраст - ${info.age}, Отдел - ${info.department}`);
}
```

```
const weatherMap = new Map();
console.log(weatherMap); // Map(0) {size: 0}

// Элементы добавляются в Map с помощью метода set():
weatherMap.set('London', '10').set('Moscow', '7');
console.log(weatherMap); // Map(2) {'London' => '10', 'Moscow' => '7'}

// Значения можно получить с помощью метода get():
console.log(weatherMap.get('Moscow')); // 7
console.log(weatherMap.get('somekey')); // undefined

// Метод has() позволяет проверить наличие ключа в Map:
console.log(weatherMap.has('Moscow')); // true
console.log(weatherMap.has('somekey')); // false

// Элементы можно удалять с помощью метода delete():
weatherMap.delete('London');
console.log(weatherMap); // Map(1) {'Moscow' => '7'}

// Полностью очистить Map можно с помощью метода clear():
weatherMap.clear();
console.log(weatherMap); // Map(0) {size: 0}

// Ключи в Map могут быть не только строками, а любым типом данных
weatherMap.set(1, 5).set(true, 'yes').set(false, 'no');
console.log(weatherMap); // Map(3) {1 => 5, true => 'yes', false => 'no'}

// При повторном применении set() с тем же ключом,
// у нас перезаписывается существующее значение
weatherMap.set(true, 'YES!').set(false, 'no?');
console.log(weatherMap); // Map(3) {1 => 5, true => 'YES!', false => 'no?'}

// В качестве ключа могут быть также массивы и объекты
weatherMap.set([1, 2, 3], 'array');
console.log(weatherMap);
// Map(4) {1 => 5, true => 'YES!', false => 'no?', Array(3) => 'array'}

weatherMap.set({ a: 1 }, { b: 1 });
console.log(weatherMap);
// Map(5) {1 => 5, true => 'YES!', false => 'no?', Array(3) => 'array', {…} => {…}}

// Размер Map можно получить с помощью свойства size:
console.log(weatherMap.size); // 5

// Чтобы мы не получили undefined при обращении к ключу массива или объекта
// нам нужно сначала создать объект или массив и указать его в качестве ключа в Map
const weatherMap2 = new Map();

const arr = [1, 2, 3];
weatherMap2.set(arr, 'array').set({ a: 1 }, 'obj');

console.log(weatherMap2.get(arr)); // array
console.log(weatherMap2.get({ a: 1 })); // undefined
```