Setters и getters

Свойства-геттеры (getters) и свойства-сеттеры (setters) - это особые методы в объектах JavaScript, которые позволяют вам получать и устанавливать значения свойств объекта. Они позволяют вам добавить логику к чтению и записи значений свойств, а также обеспечивают контроль доступа к свойствам. Давайте подробно рассмотрим, как они работают и как их использовать.

Геттеры (getters):

Геттеры - это методы, которые позволяют вам получать значение свойства объекта, как если бы оно было обычным свойством. Для создания геттера используется ключевое слово `get` внутри объекта.

Пример использования геттера:

```
const person = {
  firstName: 'John',
  lastName: 'Doe',
  get fullName() {
    return this.firstName + ' ' + this.lastName;
  }
};

console.log(person.fullName); // Выведет: "John Doe"
```

В этом примере у объекта `person` есть геттер `fullName`, который вычисляет полное имя на основе `firstName` и `lastName`. При обращении к `person.fullName`, вызывается геттер, и вы получаете полное имя.

Сеттеры (setters):

Сеттеры - это методы, которые позволяют вам устанавливать значение свойства объекта, как если бы оно было обычным свойством. Для создания сеттера используется ключевое слово `set` внутри объекта.

Пример использования сеттера:

```
const person = {
  firstName: 'John',
  lastName: 'Doe',
  set fullName(name) {
    const [first, last] = name.split(' ');
    this.firstName = first;
    this.lastName = last;
  }
};

person.fullName = 'Alice Johnson';
console.log(person.firstName); // Выведет: "Alice"
console.log(person.lastName); // Выведет: "Johnson"
```

Здесь у объекта `person` есть сеттер `fullName`, который разбивает переданное полное имя на `firstName` и `lastName` и устанавливает соответствующие значения.

Зачем использовать геттеры и сеттеры:

1. **Изолирование данных**: Геттеры и сеттеры позволяют скрыть детали реализации и изолировать данные от прямого доступа, что обеспечивает безопасность и согласованность данных.
2. **Добавление логики**: Вы можете добавить дополнительную логику при чтении или записи свойств. Например, валидацию данных или логирование.
3. **Инкапсуляция**: Геттеры и сеттеры позволяют создавать интерфейсы для работы с объектами, скрывая внутренние детали реализации.
4. **Обратная совместимость**: Если в будущем вам понадобится изменить логику чтения или записи, вы можете сделать это, не нарушая существующий код, который использует геттеры и сеттеры.

Геттеры и сеттеры делают ваш код более читаемым, гибким и безопасным. Они позволяют управлять доступом к данным и предоставляют интерфейс для работы с объектами.