Урок 3.9 - 3.10. Hoisting. Closures. Стиль кода. Объекты

Hoisting

- Всплытие объявления переменных и функций наверх контекста.
- Выполняется движком "под капотом".
- Позволяет использовать функции до их объявления.

```
// ... some code

sayHi();

// ... some code

function sayHi() {
    console.log('Hello');
}
```

JS под капотом преобразует код выше в следующиий код:

```
function sayHi() {
    console.log('Hello');
}

// ... some code

sayHi();

// ... some code
```

Замыкания

- Способность функции запоминать переменные из контекста, где она была создана

```
let func;
for (let i = 0; i < 10; i++) {
    if (i === 5) {
        func = function () {
            console.log(i);
        }
    }
}
// несмотря на то, что i здесь уже недоступна,
// при создании функции i было равно 5,
// поэтому при вызове функции выведется цифра 5.
func();
```

Создание объекта

```
// Объект со значением
const person = {
    name: 'Andrew',
    lastName: 'Gulin',
    body: {
        height: 175,
        age: 25
    }
};

// Пустой объект
const person2 = {};

// Создание свойства name у person2
person2.name = 'Борис';

// Изменение свойства age у person
person.age = 72;
```

Сравнение объектов

При использовании операторов `==` / `===` результат `true` будет только в том случае, если обе ссылки указывают на один и тот же объект

Ссылки

- При создании объека он создаётся в памяти, а переменной присваивается ссылка на этот объект.
- Если присвоить другой переменной значение текущей (ссылку на объект), то обе ссылки будут ссылаться на один и тот же объект в памяти

```
const obj = {
    property: 1,
    count: 25
};

// obj2 присваивается значение переменной obj (ссылка на объект)
const obj2 = obj;

// Изменяя obj2, мы также изменяем и obj1
obj2.property = 2; // obj.property тоже приняла значение 2
```

Доступ к свойствам

Доступ к свойствам объекта осуществляется через точку (за пример берём объект person) `console.log(person.body.age);`

Опциональная цепочка

Осуществляет проверку, есть ли текущее свойство, чтобы вызвать следующее.

За пример берём объект person2

```
// body не существует, значит body - undefined
// При попытке вызова age от undefined JS
// выдаст ошибку Cannot read property age of undefined
console.log(person2.body.age);
```

Для инлайн-проверки на существование используется опциональная цепочка

```
// Если body существует у person2 (не undefined),
// В консоль выведется значение age
// Если в body ничего нет - выведется undefined - значение свойств body
console.log(person2.body?.age);
```