**Конспект «Объекты». Раздел 2**

**Методы объекта**

В объектах могут храниться любые типы данных, в том числе и функции. Такие свойства-функции называются методами объектов. Вызов метода записывается так: объект.метод ().

let cat = {

name: 'Кекс',

color: 'рыжий',

age: 5,

getGreeting: function() {

return 'Мяу, привет!';

}

};

console.log(cat.getGreeting()); // Выведет: Мяу, привет!

**Скобочная нотация**

Прочитать свойство из объекта можно с помощью квадратных скобок: catsFavoriteFood[name]. Способ со скобками называется *скобочной* нотацией, способ с точкой — *точечной* нотацией.

Скобочная нотация намного гибче точечной. Например, вы можете прочитать из объекта свойство, название которого записано в переменную:

let name = 'Кекс';

let catsFavoriteFood = { 'Кекс': 'рыба' };

console.log(catsFavoriteFood.name); // Выведет: undefined

console.log(catsFavoriteFood[name]); // Выведет: рыба

Первое сообщение содержит undefined, потому что у объекта нет свойства name. Второе сообщение содержит искомое значение, потому что программа понимает, что в квадратных скобках переменная. Значение переменной подставится в скобки, а затем будет найдено нужное свойство объекта.

В качестве ключей в объекте можно использовать любые строки, даже строки с пробелами. С точечной нотацией такие свойства прочитать не получится, а со скобочной — без проблем.

let cat = { 'favorite food': 'Сметана' };

// Вызовет ошибку

console.log(cat.favorite food);

// Отработает нормально

console.log(cat['favorite food']);

**Объект как словарь**

Словари, или мапы, очень удобны в использовании. В нашем примере они хранят соотношение имени кота и лакомства, которое по вкусу именно ему.

let catsFavoriteFood = {

Кекс: 'рыба',

Рудольф: 'котлета',

Снежок: 'сметана'

};

let printFavoriteFood = function (name) {

// Используем скобочную нотацию

return 'Моя любимая еда — ' + **catsFavoriteFood[name]**;

};

console.log(printFavoriteFood('Снежок')); // Выведет: Моя любимая еда — сметана

**this**

Изнутри методов можно обращаться к свойствам и другим методам объекта с помощью ключевого слова this. Оно указывает на текущий объект и называется *контекстом вызова*.

Важная деталь: пока функция не вызвана, this не содержит никакого значения, контекст появляется только в момент вызова функции.

let cat = {

name: 'Кекс',

color: 'рыжий',

age: 5,

getGreeting: function() {

return 'Мяу, привет! Меня зовут ' + this.name;

}

};

console.log(cat.getGreeting()); // Выведет: Мяу, привет! Меня зовут Кекс