1. Объект – это структура данных, представляющая собой набор свойств и их значений, а также специальных методов для их обработки. Объекты являются одним из типов данных JavaScript. Например:

let car = {

name: 'Ford',

model: 'Mustang',

year: 2019

};

2. Существует несколько способов создания объектов: при помощи литерала объекта, функции-конструктора, класса, а также с помощью встроенных в JavaScript конструкторов (Object, Array, Date и т.д.).

3. Литерал объекта – это синтаксис для создания объекта в JavaScript. Он заключается в фигурные скобки {} и содержит пары ключ-значение через запятые:

let car = {

name: 'Ford',

model: 'Mustang',

year: 2019

};

4. Нет, объект, который был объявлен при помощи директивы const, изменить нельзя. Директива const предназначена для объявления констант и для присвоения значений неизменяемых переменных.

5. Вычисляемые свойства – это свойства, значение которых может быть вычислено во время выполнения программы. Они могут быть заданы при помощи литерала объекта, а значение определяется в форме выражения. Например:

let car = {

name: 'Ford',

model: 'Mustang',

year: 2019,

getAge: function() {

let currentYear = new Date().getFullYear();

return currentYear - this.year;

}

};

6. Короткие свойства – это свойства, имя которых состоит из одного слова. Они могут быть заданы при помощи литерала объекта, используя синтаксис короткой записи. Например:

let car = {

name,

model,

year

};

7. Тип данных свойств объекта может быть любым типом данных JavaScript, включая числа, строки, логические значения и другие объекты.

8. Оператор new используется для создания экземпляра объекта при помощи функции-конструктора. Функция-конструктор – это функция, которая используется для создания объекта и инициализации его свойств. Например:

function Car(name, model, year) {

this.name = name;

this.model = model;

this.year = year;

}

let car = new Car('Ford', 'Mustang', 2019);

9. К свойствам объекта можно обращаться при помощи точки или квадратных скобок. Если имя свойства состоит из нескольких слов, то для обращения к нему используется только квадратные скобки. Например:

let car = {

name: 'Ford',

model: 'Mustang',

year: 2019

};

// Обращение к свойству name с помощью точки

console.log(car.name); // Ford

// Обращение к свойству model с помощью квадратных скобок

console.log(car['model']); // Mustang

// Обращение к свойству year с помощью квадратных скобок

let propName = 'year';

console.log(car[propName]); // 2019

10. Свойство prototype – это свойство, которое позволяет добавлять или изменять методы и свойства объекта. Это полезно в случаях, когда нужно использовать один и тот же набор методов для нескольких экземпляров объекта.

11. Существует несколько способов добавления свойств для объекта: с помощью литерала объекта, присваиванием значения или функции, использованием оператора new или динамически созданными свойствами.

12. Свойство объекта можно удалить с помощью оператора delete. Например:

delete car.name;

13. Тип данных свойства объекта может быть любым типом данных JavaScript, включая числа, строки, логические значения и другие объекты.

14. Есть несколько способов добавления методов к объектам в JavaScript: через присваивание свойству объекта (например, obj.myMethod = function(){...}), через расширение прототипа объекта (например, Object.prototype.myMethod = function(){...}), используя метод Object.defineProperty() (например, Object.defineProperty(obj, 'myMethod', {value: function(){...}, enumerable: false}).

15. Объект-прототип — это свойство, которое определяет поведение и атрибуты объекта. Оно используется для поиска свойств и методов для объекта, если они не были найдены в конкретном объекте. Например, все объекты в JavaScript наследуют прототип от объекта Object. Это значит, что все объекты могут использовать методы и свойства объекта Object, такие как toString(), hasOwnProperty(), и т.д.

16. Есть несколько способов проверки существования свойства в объекте: через проверку с помощью оператора in (например, 'prop' in obj), через проверку с помощью метода hasOwnProperty (например, obj.hasOwnProperty('prop')), через проверку с помощью метода getOwnPropertyDescriptor (например, Object.getOwnPropertyDescriptor(obj, 'prop')).

17. Ключевое слово this в JavaScript является ссылкой на объект, в котором вызывается текущий контекст. Оно может быть использовано для обращения к свойствам и методам объекта или для вызова метода в рамках текущего объекта. Например, в следующем примере this ссылается на объект person:

var person = {

firstName: 'John',

lastName: 'Doe',

fullName: function() {

return this.firstName + ' ' + this.lastName;

}

};

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

18. Для проверки и определения типа данных в JavaScript можно использовать следующие методы и свойства: свойство typeof (например, typeof x === 'string'), функция instanceof (например, x instanceof Object === true), метод Object.prototype.toString (например, Object.prototype.toString.call(x) === '[object String]').