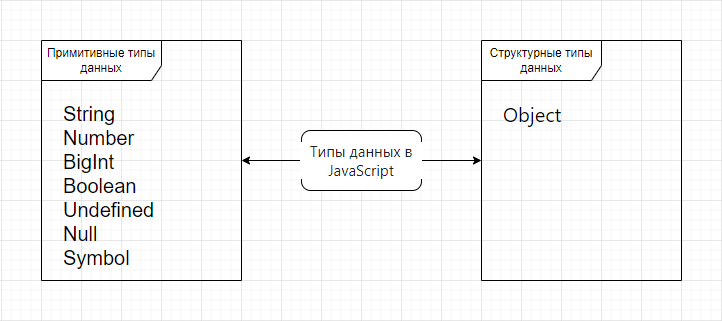
1. **Основы JavaScript:**
   * Каковы типы данных есть в JavaScript? Составить диаграмму.(Figma, diagrams.net и аналогичные)



* + В чем разница между let, const и var при объявлении переменных?

var: имеет функциональную область видимости, может быть переопределена и обновлена.

let: имеет блочную область видимости, не может быть повторно объявлена, но может быть обновлена.

const: также имеет блочную область видимости, не может быть переопределена и обновлена.

* + Что такое hoisting?

Hoisting - это механизм в JavaScript, при котором объявления переменных и функций "поднимаются" вверх своей области видимости перед выполнением кода. Это позволяет использовать переменные и функции до их фактического объявления. Однако только объявления "поднимаются", а присвоения остаются на месте.

1. **Функции:**
   * Как объявить функцию в JavaScript?

С помощью function declaration

* + В чем разница между function declaration и function expression?

Разница между function declaration и function expression заключается в том, что function declaration поднимается (hoisted) вверх области видимости, что позволяет вызывать функцию до её объявления. В то время как function expression не поднимается и должна быть объявлена перед использованием.

* + Что такое стрелочные функции?

Стрелочные функции (arrow functions) - это новый синтаксис объявления функций в JavaScript, который появился в ECMAScript 6. Они предоставляют более краткую и читаемую запись для определения функций. Стрелочные функции не имеют собственного this и привязывают его к внешнему контексту, что делает их часто удобными в использовании.

1. **Объекты и массивы:**
   * Как создать объект в JavaScript?

Чтобы создать объект в JavaScript, вы можете использовать следующий синтаксис:

let object = {

key1: value1,

key2: value2,

};

* + Как добавить новый элемент в конец массива?

Чтобы добавить новый элемент в конец массива, вы можете использовать метод push():

let array = [1, 2, 3];

array.push(4);

* + Как перебрать все элементы массива?

Чтобы перебрать все элементы массива, вы можете использовать цикл for или метод forEach():

let array = [1, 2, 3, 4];

for (let i = 0; i < array.length; i++) {

console.log(array[i]);

}

array.forEach(function(element) {

console.log(element);

});

1. **Обработка событий:**
   * Как назначить обработчик события элементу?

Назначить обработчик события элементу можно с помощью метода addEventListener().

* + Что такое event delegation?

Event delegation (делигирование событий) — это техника обработки событий, при которой один обработчик события назначается родительскому элементу, чтобы обрабатывать события от его потомков. Это позволяет избежать назначения обработчика каждому элементу отдельно и улучшить производительность.

* + Как отменить стандартное поведение события?

Чтобы отменить стандартное поведение события, можно использовать метод preventDefault() объекта события.

1. **Асинхронность:**
   * Как работает синхронный и асинхронный код в JavaScript?

Синхронный код в JavaScript выполняется по порядку, одна операция завершается, затем выполняется следующая. Если в коде встречается блокирующая операция, то весь код будет приостановлен до её завершения.

Асинхронный код в JavaScript позволяет выполнять операции параллельно. Когда встречается асинхронная операция, она запускается в фоновом режиме, а выполнение кода продолжается дальше.

* + Что такое Promise?

Promise - это объект, представляющий какое-то значение, которое может быть доступно сразу, в будущем или никогда. Он позволяет выполнять асинхронный код и управлять последовательностью его выполнения.

* + Как использовать async/await?

Async/await - это синтаксический сахар, который упрощает работу с промисами. Ключевое слово async используется для обозначения асинхронной функции, которая всегда возвращает промис. Ключевое слово await используется внутри асинхронной функции для приостановки выполнения кода до тех пор, пока промис не будет разрешен. Это позволяет писать асинхронный код таким образом, будто он синхронный, что делает его более читаемым и понятным.