**Операторы JS+4 способа внедрения скриптов**

**Введение в JS**

1. **Назначение языка программирования JavaScript** JavaScript (JS) — это язык программирования, который используется для создания динамически обновляемого контента, управления мультимедиа, анимации и многое другое на веб-страницах. Его основное назначение — взаимодействие с пользователем и улучшение пользовательского интерфейса путем добавления интерактивности на веб-страницы.
2. **Актуальная последняя версия языка JS** Последняя версия языка JavaScript известна как ECMAScript 2023 (ES14). Стандарт ECMAScript обновляется ежегодно, и версия 2023 года является самой актуальной на данный момент.
3. **Для разработки каких элементов веб-сайта используют скрипты на JS? Примеры** Скрипты на JavaScript используются для разработки следующих элементов веб-сайта:

* **Формы**: проверка данных формы перед отправкой.
* **Галереи изображений**: создание слайд-шоу и динамических галерей.
* **Анимации**: анимация элементов на странице, например, плавные переходы.
* **Модальные окна**: создание всплывающих окон с информацией.
* **Интерактивные карты**: использование картографических сервисов для отображения карт и добавления меток.
* **Чат-боты**: создание интерфейсов для общения с пользователями.

1. **Сходство языка программирования JavaScript с языком Java** JavaScript и Java имеют сходство лишь в названии и некотором синтаксисе, однако они сильно различаются по своему назначению и использованию:

* **Сходство**: Оба языка используют синтаксис, основанный на C-подобных конструкциях (например, синтаксис для циклов и условных операторов).
* **Различия**: JavaScript — это интерпретируемый язык, работающий в браузере (на стороне клиента), а Java — компилируемый язык, обычно используемый для серверных приложений. У них разная модель выполнения, управление памятью и экосистема.

1. **Название редактора кода для скриптов на JS, в котором вы планируете работать** Один из популярных редакторов кода для работы с JavaScript — Visual Studio Code (VS Code). Он предоставляет мощные возможности для работы с кодом, включая поддержку плагинов, автодополнение кода и встроенные терминалы.
2. **Пример синтаксиса скрипта на JS**

javascript

Копировать код

// Пример простого скрипта на JavaScript

function greet(name) {

console.log(`Привет, ${name}!`);

}

greet("Мир"); // Вывод: Привет, Мир!

1. **Примеры на 4 разных способа внедрения JS в HTML код**

* **Внутри тега <script>**

html

Копировать код

<!DOCTYPE html>

<html>

<head>

<title>Пример</title>

</head>

<body>

<script>

document.write("Привет, мир!");

</script>

</body>

</html>

* **Внешний файл**

html

Копировать код

<!DOCTYPE html>

<html>

<head>

<title>Пример</title>

<script src="script.js"></script>

</head>

<body>

<!-- Контент страницы -->

</body>

</html>

javascript

Копировать код

// script.js

document.write("Привет, мир!");

* **Событийные атрибуты (inline event handlers)**

html

Копировать код

<!DOCTYPE html>

<html>

<head>

<title>Пример</title>

</head>

<body>

<button onclick="alert('Привет, мир!')">Нажми меня</button>

</body>

</html>

* **Использование функции внутри HTML**

html

Копировать код

<!DOCTYPE html>

<html>

<head>

<title>Пример</title>

<script>

function showMessage() {

alert('Привет, мир!');

}

</script>

</head>

<body>

<button onclick="showMessage()">Нажми меня</button>

</body>

</html>

**Синтаксис JS**

* 1. **Типы данных в языке программирования JavaScript** В JavaScript можно работать с несколькими основными типами данных:
* **Примитивные типы данных**:
  + **Number**: целые и вещественные числа.
  + **String**: строки.
  + **Boolean**: логические значения (true/false).
  + **Undefined**: неопределенное значение.
  + **Null**: значение отсутствия объекта.
  + **Symbol**: уникальные и неизменяемые значения.
  + **BigInt**: целые числа произвольной точности.
* **Объектные типы данных**:
  + **Object**: коллекция свойств и методов.
  + **Array**: массивы.
  + **Function**: функции.
  + **Date**: даты.
  + **RegExp**: регулярные выражения.

Примеры описания переменных целого, вещественного и строкового типов данных:

javascript

Копировать код

// Целое число

let wholeNumber = 42;

// Вещественное число

let floatNumber = 3.14;

// Строка

let greeting = "Привет, мир!";

* 1. **Операции в языке JavaScript** В JavaScript используются различные операции, среди которых:
* **Арифметические операции**: +, -, \*, /, %, \*\*

javascript

Копировать код

let sum = 5 + 3; // сложение: 8

let difference = 5 - 3; // вычитание: 2

let product = 5 \* 3; // умножение: 15

let quotient = 5 / 3; // деление: 1.666...

let remainder = 5 % 3; // остаток от деления: 2

let power = 5 \*\* 3; // возведение в степень: 125

* **Операции сравнения**: ==, ===, !=, !==, >, <, >=, <=

javascript

Копировать код

let isEqual = (5 == '5'); // true (сравнение по значению)

let isStrictEqual = (5 === '5'); // false (строгое сравнение)

let isNotEqual = (5 != '5'); // false

let isStrictNotEqual = (5 !== '5'); // true

let isGreater = (5 > 3); // true

let isLess = (5 < 3); // false

let isGreaterOrEqual = (5 >= 5); // true

let isLessOrEqual = (5 <= 3); // false

* **Логические операции**: &&, ||, !

javascript

Копировать код

let andOperation = (true && false); // false

let orOperation = (true || false); // true

let notOperation = (!true); // false

* **Операции присваивания**: =, +=, -=, \*=, /=, %=

javascript

Копировать код

let x = 10;

x += 5; // x = x + 5; x равно 15

x -= 3; // x = x - 3; x равно 12

x \*= 2; // x = x \* 2; x равно 24

x /= 4; // x = x / 4; x равно 6

x %= 5; // x = x % 5; x равно 1

* **Тернарная операция**: condition ? expr1 : expr2

javascript

Копировать код

let age = 18;

let canVote = (age >= 18) ? 'Да' : 'Нет'; // canVote равно 'Да'

* 1. **Пример использования методов alert(…), confirm(…) и prompt() объекта window для вывода окон сообщений**

javascript

Копировать код

// alert(…)

alert("Привет, мир!"); // Показать простое окно с сообщением

// confirm(…)

let isConfirmed = confirm("Вы уверены?");

if (isConfirmed) {

alert("Вы нажали 'OK'");

} else {

alert("Вы нажали 'Отмена'");

}

// prompt(…)

let name = prompt("Как вас зовут?", "Имя по умолчанию");

if (name !== null) {

alert(`Привет, ${name}!`);

} else {

alert("Вы не ввели имя.");

}

**Операторы JS**

* 1. **Условный оператор (if/else)**

Условный оператор if/else используется для выполнения определенного блока кода в зависимости от истинности условия.

**Синтаксис:**

javascript

Копировать код

if (условие) {

// блок кода, выполняемый, если условие истинно

} else if (другое\_условие) {

// блок кода, выполняемый, если другое\_условие истинно

} else {

// блок кода, выполняемый, если ни одно из условий не истинно

}

**Примеры:**

javascript

Копировать код

let age = 20;

if (age < 18) {

console.log("Вы слишком молоды.");

} else if (age >= 18 && age < 65) {

console.log("Вы взрослый.");

} else {

console.log("Вы пенсионер.");

}

* 1. **Оператор множественного выбора (switch)**

Оператор switch используется для выбора одного из множества возможных блоков кода для выполнения в зависимости от значения выражения.

**Синтаксис:**

javascript

Копировать код

switch (выражение) {

case значение1:

// блок кода, выполняемый, если выражение === значение1

break;

case значение2:

// блок кода, выполняемый, если выражение === значение2

break;

// другие case

default:

// блок кода, выполняемый, если ни одно из значений не совпало

}

**Пример:**

javascript

Копировать код

let day = 3;

switch (day) {

case 1:

console.log("Понедельник");

break;

case 2:

console.log("Вторник");

break;

case 3:

console.log("Среда");

break;

default:

console.log("Неизвестный день");

}

* 1. **Операторы организации циклов (for, while, do…while)**

**Синтаксис и примеры:**

* **for**

javascript

Копировать код

for (инициализация; условие; итерация) {

// блок кода, выполняемый на каждой итерации

}

// Пример:

for (let i = 0; i < 5; i++) {

console.log(i); // Выводит числа от 0 до 4

}

* **while**

javascript

Копировать код

while (условие) {

// блок кода, выполняемый, пока условие истинно

}

// Пример:

let i = 0;

while (i < 5) {

console.log(i); // Выводит числа от 0 до 4

i++;

}

* **do…while**

javascript

Копировать код

do {

// блок кода, выполняемый хотя бы один раз, и затем повторяемый, пока условие истинно

} while (условие);

// Пример:

let i = 0;

do {

console.log(i); // Выводит числа от 0 до 4

i++;

} while (i < 5);

* 1. **Оператор досрочного выхода из циклов (break)**

Оператор break используется для немедленного выхода из цикла или оператора switch.

**Пример:**

javascript

Копировать код

for (let i = 0; i < 10; i++) {

if (i === 5) {

break; // Прерывает цикл, когда i равно 5

}

console.log(i); // Выводит числа от 0 до 4

}

* 1. **Оператор продолжения (continue)**

Оператор continue используется для пропуска текущей итерации цикла и перехода к следующей.

**Пример:**

javascript

Копировать код

for (let i = 0; i < 10; i++) {

if (i % 2 === 0) {

continue; // Пропускает текущую итерацию, если i - четное число

}

console.log(i); // Выводит только нечетные числа

}

* 1. **Оператор возврата (return)**

Оператор return используется для завершения выполнения функции и возврата значения.

**Синтаксис:**

javascript

Копировать код

return выражение;

**Пример:**

javascript

Копировать код

function sum(a, b) {

return a + b; // Возвращает сумму a и b

}

let result = sum(5, 3);

console.log(result); // Выводит: 8