1. Что такое HTML?

HTML расшифровывается как Hyper Text Markup Language. Это язык разметки, состоящий из различных тегов. Используется для определения структуры веб-страниц.

2. Что такое семантические элементы в HTML?

В семантические элементы в HTML - это элементы, которые содержат смысл содержимого и структуру HTML-документа. Эти элементы содержат контент, связанный с их названиями или отражающий их названия. Вот некоторые из семантических HTML-элементов, перечисленных ниже:

Заголовок

Главная

Раздел

Статья

В сторону

Нижний колонтитул и т.д.

3. Что такое пустые элементы в HTML?

В пустые элементы в HTML - это элементы, для которых не требуется закрывающий тег, за которым следует открывающий тег. Эти элементы также известны как самозакрывающиеся элементы. Пример: <img>, <input>, <br>, <hr> и т.д.

4. Проводите различие между встроенными и блочными элементами в HTML.

Встроенные элементы в HTML есть элементы, которые не начинаются каждый раз с новой строки и не занимают столько же места и ширины, сколько занимает контент. Свойства полей и отступов, примененные вверху и внизу этих элементов, могут не дать ожидаемого эффекта. Примеры: <a>, <strong>, <img>, <input> и т.д.

Элементы блока автоматически начинаются с новой строки и занимают всю ширину окна просмотра независимо от содержащегося содержимого. Свойства padding и margin оказывают одинаковое влияние на все четыре стороны. Примеры: <div>, <h1> - <h6>, <p>, <таблица> и т.д.

5. Что такое list в HTML? Объясните различные типы списков, доступные в HTML.

В HTML Списки используются для представления набора различных элементов. Доступны три типа списков в формате HTML, перечисленных ниже:

1. Неупорядоченный список: Он определяется с помощью тегов <ul> и <li>. По умолчанию элементы отмечены маркированной точкой.

<ul>

<li>List Item 1</li>

<li>List Item 3</li>

<li>List Item 3</li>

</ul>

2. Упорядоченный список: Он определяется с помощью тегов <ol> и <li>. По умолчанию он представляетСписок элементы с цифровыми обозначениями.

<ol>

<li>List Item 1</li>

<li>List Item 3</li>

<li>List Item 3</li>

</ol>

3. Список определений: Это особый вид списка, который используется для перечисления терминов or с их определениями. Его можно определить с помощью тегов <dl>, <dt> и <dd>. dt – термин определения, dd – описание определения

<dl>

<dt>First term</dt>

<dd>Definition 1</dd>

<dt>Second term</dt>

<dd>Definition 2</dd>

<dt>Third term</dt>

<dd>Definition 3</dd>

</dl>

6. Какова базовая структура HTML-документа?

Базовая структура HTML-документа в HTML5 показана ниже:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Document</title>

</head>

<body></body>

</html>

7. Объясните элементы, используемые в базовой структуре HTML-документа.

Элементы, используемые в базовой структуре HTML-документа, объясняются ниже:

<!DOCTYPE html>: Он представляет HTML5-версию HTML.

<html>: Это корневой элемент HTML-документа.

<заголовок>: Оно содержит метаданные, то есть сведения о данных. Содержимое, содержащееся в этом теге, не отображается на веб-странице.

<название>: Оно содержит заголовок документа, который будет виден на вкладке браузера.

<основная часть>: Оно содержит содержимое веб-страницы в виде HTML-тегов, таких как div, анкор, абзац, заголовки и т.д.

8. Объясните теги в HTML.

Теги HTML используются для определения элементов веб-страницы. По сути, это ключевые слова, заключенные в угловые скобки (<>). Примерами HTML-тегов являются <div>, <p>, <a>, , <img> и т.д.

9. Почему атрибут ‘alt’ используется с Тег<img> в HTML?

Альтернативный атрибут предоставляет альтернативный контент, связанный с изображением, который будет показан на веб-странице, если изображение не будет загружено. В этом случае он используется для определения нашего изображения.

10. В чем разница между <div> и тег в HTML?

В приведенной ниже таблице показаны различия между тегами div и span в HTML:

тег <div>

<span> тег

Это элемент блочного уровня.

Это встроенный элемент.

Его можно использовать для группировки и структурирования содержимого веб-страницы.

В основном используется для взаимодействия и оформления определенной части веб-страницы.

Это представляет собой больший раздел веб-страницы.

Используется для таргетинга небольших частей веб-страницы.

Оно начинается с новой строки и занимает всю доступную ширину.

Оно не начинается с новой строки и занимает только необходимую ширину в соответствии с содержанием.

11. Почему используется тег <meta charset = “UTF-8”>?

Используется для установки кодировки символов в документе на UTF-8 как правильно отобразить текст и специальные символы на веб-странице.

12. С какой целью используется атрибут ‘role’ в HTML?

The Рольатрибут определяет функциональность и назначение элемента, в основном доступность. В нем представлена дополнительная информация о вспомогательных технологиях, таких как программы чтения с экрана, чтобы донести точное значение элемента до пользователей с ограниченными возможностями.

13. Проводите различие между методами GET и POST в HTML-формах.

В приведенной ниже таблице объясняются различия между методами GET и POST в HTML-формах:

Метод GET

Метод ПУБЛИКАЦИИ

Это небезопасный способ отправки данных на сервер.

Это безопасный способ отправки данных формы.

Все параметры данных формы видны в URL.

Ни один из параметров нигде не виден.

У него есть ограничение по длине URL, которое варьируется для разных браузеров.

У него большее ограничение по длине URL по сравнению с ограничением метода GET.

Результаты по умолчанию кэшируются браузером.

Ответы по умолчанию не кэшируются в браузере.

Пользователи могут добавлять в закладки отправленную форму.

Ответы можно легко добавить в закладки.

14. Для чего нужен тег <iframe>?

Тег <iframe> используется для встраивания внешних документов или веб-страниц внутрь текущего документа путем указания ссылки внутри него. В основном используется для встраивания внешних видео, карт и другого внешнего контента.

15. Объясните особенности HTML5?

В HTML5 появились некоторые новые функции, которые перечислены ниже:

Введены новые семантические элементы, такие как: <заголовок>, <нижний колонтитул>, <Навигация>, <в сторону>, <статья>, <раздел> и т.д.

Новые типы ввода формы, такие как адрес электронной почты, URL, номер, дата и т.д.

В нем были представлены <аудио> и <видео> теги для встраивания аудио и видео и снижения зависимости от сторонних библиотек.

<холст> элемент для рисования графики и анимации с использованием JavaScript.

Представляет хранилище браузера как localStorage и Хранилище сеансов для хранения данных в браузере.

16. Что такое localStorage?

localStorage - это механизм веб-хранилища на стороне клиента, который позволяет веб-приложениям постоянно хранить пары ключ-значение в веб-браузере пользователя. Он предоставляет простой интерфейс для локального хранения данных.

17. Что такое sessionStorage?

Это также API веб-хранилища, предоставляемый веб-браузерами для хранения данных аналогично тому, как они хранятся в localStorage в виде пар ключ-значение. Данные, хранящиеся в Хранилище сеансов будут доступны только в течение одного сеанса, так что, если пользователь закроет окно или вкладку, сохраненные данные будут потеряны.

18. Проводите различие между localStorage и sessionStorage.

В приведенной ниже таблице перечислены различия между локальным и сессионным хранилищем:

localStorage

Хранилище сеансов

Сохраненные данные могут быть доступны на всех открытых вкладках браузера.

Доступ к сохраненным данным ограничен только текущей вкладкой или окном.

Данные будут постоянными, если вкладки будут закрыты и открыты снова.

Данные будут потеряны, как только пользователь закроет вкладку или окно.

Используется для долгосрочного хранения данных.

Он используется для хранения данных столько, сколько вы захотите.

Данные можно очистить с помощью очистить() только методом.

Данные можно очистить, либо закрыв вкладку, либо используя очистить() Также метод.

19. Какова цель использования элементов <figure> и <figcaption> в HTML5?

Элемент <рисунок> используется для отображения медиаконтента на веб-странице, такого как аудио, видео и т.д. В то время как элемент <figcaption> используется для подписи или легенды к контенту, отображаемому элементом <рисунок> .

20. Напишите HTML-код для создания таблицы с 3 столбцами и 3 строками.

Приведенный ниже код создает таблицу с 3 строками и 3 столбцами:

<table border="1px">

<thead>

<tr>

<th>col 11</th>

<th>col 12</th>

<th>col 13</th>

</tr>

</thead>

<tbody>

<tr>

<td>col 21</td>

<td>col 22</td>

<td>col 23</td>

</tr>

<tr>

<td>col 31</td>

<td>col 32</td>

<td>col 33</td>

</tr>

</tbody>

</table>

21. Как можно объединить строки и столбцы HTML-таблицы?

Вы можете использовать атрибуты colspan и rowspan с элементом <td> и указать количество строк и столбцов, которые будут объединены, передав числовое значение определенным атрибутам. Атрибут colspan можно использовать для объединения столбцов, а атрибут rowspan - для объединения строк.

22. Опишите атрибуты данных в HTML.

В атрибуты данных в HTML определены для сбора данных, которые являются частными для веб-страницы или веб-приложения. В основном он используется для персонализации JavaScript для конкретного пользователя или посетителя на веб-странице на основе действий этого пользователя на веб-странице. В настоящее время атрибуты данных рекомендуется не использовать на веб-странице, поскольку эти атрибуты можно легко изменить, перейдя в браузерСредства разработки и проверьте элемент.

23. Что такое CSS?

CSS, который расшифровывается как каскадные таблицы стилей. Он помогает спроектировать и стилизовать веб-страницу, чтобы сделать ее привлекательной для пользователей. CSS предоставляет нам множество селекторов для выбора HTML-элементов и оформления их в соответствии с требованиями. Некоторые из CSS-селекторов являются Селекторами элементов, Селекторы классов и Селекторы идентификаторов.

24. Объясните селекторы в CSS.

В CSS селекторы используются для выбора элементов и оформления элемента путем предоставления ему свойств CSS. Ниже приведен список некоторых распространенных CSS-селекторов:

Выбор элемента: Выберите напрямую, используя название элемента.

Выбор идентификатора: Определите атрибут ID и выберите, используя префикс #, за которым следует значение атрибута ID.

Выбор класса: Определите атрибут класса и выберите, используя . префикс, за которым следует значение атрибута класса.

Универсальный селектор (\*): Выберите с помощью знака \*.

Селектор атрибутов: Выберите элементы на основе значений атрибутов. Например: ввод [type=”text”]{}

Прямой дочерний селектор: Выберите элемент с помощью любого из вышеперечисленных селекторов и используйте >, за которым следует прямой дочерний селектор. Например: родитель > потомок{}

Псевдоселекторы: Это такие селекторы, как:hover, :nth-child(), ::after, ::before и т.д.

25. Объясните приоритет селекторов классов, идентификаторов и элементов в CSS.

Приоритет CSS-селекторов ID, Class и Element показан ниже:

Выбор идентификатора > Выбор класса> Выбор элемента

Выбор идентификатора + выбор класса > Выбор идентификатора + выбор элемента > Выбор класса + выбор элемента

26. Каковы наилучшие практики использования JavaScript и CSS?

Лучшие практики для JavaScript и CSS могут быть определены в соответствии с требованиями проекта. Ниже перечислены некоторые общие рекомендации для JavaScript и CSS:

Всегда используйте внешний файл для определения JavaScript и CSS с помощью .js и .css расширения соответственно.

Всегда связывайте CSS-файл внутри тега <head> HTML-документа.

Всегда добавляйте файл сценария в конец тега <body> непосредственно перед тем местом, где закрывается body.

Попробуйте писать JavaScript в строгом режиме, чтобы избежать ошибок и написать более чистый JavaScript.

Избегайте создания глобальной переменной и используйте значимое имя для определения переменных.

27. В чем разница между свойствами visibility: hidden и display: none в CSS?

Свойство видимость: скрыто скрывает только содержимое элемента, в котором оно используется. Элемент не удаляется из документа, а пространство остается таким, какое оно есть, чтобы никакой другой элемент не мог заменить его в пользовательском интерфейсе. С другой стороны, отображение: отсутствует свойство не только скрывает элемент, но и удаляет его из документа, а пространство, занимаемое элементом, теперь свободно для использования другими элементами.

28. Упоминаете проблемы, с которыми сталкиваются разработчики при запуске CSS в Internet Explorer (IE)?

В приведенном ниже списке показаны проблемы, с которыми сталкиваются разработчики в Internet Explorer:

Прозрачность изображений с .png расширение.

Проблемы, связанные со свойством Z-index.

Иногда это удваивает поле, добавляемое к элементу.

Коробочная модель имеет другую интерпретацию.

Отсутствует поддержка функций CSS3.

29. Объясните box-модель в CSS.

Прямоугольная модель в CSS - это, по сути, предварительный шрифт элемента с некоторыми свойствами. Прямоугольная модель содержит четыре элемента: содержимое, заполнение, граница и margin .

Контент: Это может быть текстовое содержимое или вложенные HTML-элементы с некоторым содержимым внутри элемента.

Заполнение: Это пространство вокруг содержимого элемента или пространство между содержимым и границами.

Граница: Это обводка или контур, придаваемый элементу для отображения его границ или придания ему стиля.

Поле: Это пространство вокруг всего элемента, это промежуток за пределами элемента от других элементов или пространство между границей этого элемента и другими элементами.

30. Для чего используется свойство z-index в CSS?

Свойство z-index используется для управления порядком размещения элементов, которые размещаются с помощью позиция свойство в CSS. При большем или положительном значении элемент будет отображаться поверх других элементов, в то время как при отрицательном или меньшем значении они будут отображаться позади элементов с более высокими значениями.

31. Как используется свойство float?

Свойство float позволяет устанавливать дочерние элементы контейнера либо с левой, либо с правой стороны от него. Возможные значения для этого свойства: left, right, initial, inherit, и Нет.

32. Объясните различные способы отображения элемента в центре веб-страницы.

Существует множество способов центрирования элемента на веб-странице, как описано ниже:

Использование поля: Свойство margin можно использовать для центрирования элемента по горизонтали, задав поле автоматически слева и справа от элемента как поле: 0 автоматически;.

Использование display: Свойство display со значением Гибкий может использоваться для центрирования элемента как по вертикали, так и по горизонтали с помощью некоторых дополнительных свойств, таких как выравнивание элементов: по центру; и обоснование-содержание: в центре;.

Используя position и transform: Вы можете расположить элемент так, чтобы придать ему левую сторонуи НаверхДля50% а затем используйте преобразование недвижимость со стоимостью Перевод (-50%);

33. Опишите использование свойства position в CSS.

Свойство position используется для позиционирования элемента относительно других элементов в соответствии с заданным значением свойства. Возможные значения этого свойства: относительный, абсолютный, фиксированный, липкий, и статичный.

34. Что такое псевдоклассы и псевдоэлементы в CSS?

Псевдоклассы и псевдоэлементы - это разные объекты в CSS. В CSS они известны как псевдоселекторы. Ниже приводится их объяснение.:

псевдоклассы: Это классы, которые выбирают элементы на основе их состояния и позиции. Некоторые псевдоклассы : наведение курсора, : фокус, : n-й дочерний элемент и т.д.

псевдоэлементы: Это виртуальные элементы, которые в основном определяются для оформления определенной части элемента в HTML-документе. Некоторые псевдоэлементы : до и : после.

35. Почему ‘!important’ используется в CSS?

The !важно объявление используется для присвоения более высокого приоритета свойству CSS, чтобы переопределить его среди других конфликтующих стилей, определенных в элементе с использованием того же свойства, например ширина: 30 пикселей!важно; переопределит свойство ширина: 25 пикселей; определены в том же элементе.

36. Для чего используется свойство ‘box-sizing’ в CSS?

Свойство размер поля используется для определения способа вычисления высоты и ширины элемента. Он определяет, будут ли включены граница и отступы для расчета высоты и ширины элемента. Общими значениями являются поле содержимого (по умолчанию) и рамка.

37. Расскажите мне о препроцессорах CSS и их преимуществах перед чистым CSS?

Препроцессор CSS создается с использованием языка сценариев, который расширяет правила CSS, и обрабатывается в обычном CSS путем создания файла с .css расширение. Чаще всего используются CSS-препроцессоры SASS и Меньше. SASS основан на языке Ruby в то время как Меньшеоснован на JavaScript, изначально он также был основан на Ruby но теперь переведен на JavaScript. Они предоставляют некоторые дополнительные функции, такие как создание переменных, миксинов, вложение кода и некоторые другие функции для повышения удобства сопровождения кода.

38. Что означает слово ‘Каскадирование’ в таблице каскадных стилей?

Каскадирование представляет собой специфический порядок применения стилей. Эти стили могут быть определены пользователем, автором или они могут быть стилями браузера по умолчанию. Специфичный порядок для стилей является пользовательские стили > авторские стили > стили браузера по умолчанию.

39. Объясните медиа-запросы в CSS.

Медиазапросы - это блок CSS-кода, определенный для определенной ширины или диапазона ширины. Их можно определить с помощью @media ключевое слово с screen для указания стилей для определенной ширины или диапазона ширины. Они очень часто используются для создания адаптивного дизайна.

40. Опишите CSS-спрайты и их важность для повышения производительности веб-сайта.

CSS-спрайты - это метод, который используется для сжатия нескольких изображений, доступных на веб-странице, в один файл изображения. Он упорядочивает все изображения в виде сетки. background-position свойство CSS можно использовать для отображения разных частей объединенного изображения в качестве фона для разных элементов. Это повышает производительность веб-сайта за счет сокращения запросов сервера, поскольку веб-сайт теперь должен запрашивать только одно изображение вместо запроса нескольких изображений.

41. Как вы можете оптимизировать загрузку CSS-файлов в браузере?

Для оптимизации загрузки CSS-файлов доступно несколько методов, перечисленных ниже:

За счет минимизации количества CSS-файлов.

Минимизация CSS

Используя кэш браузера

Загрузите ненужные стили, используя асинхронную или дифференцированную загрузку.

42. Как создавать адаптивный дизайн?

В CSS доступны несколько ключевых концепций, которые могут помочь вам в создании адаптивного дизайна, как указано ниже:

Использование Медиа-запросы

Использование макет flexbox

Использование Макет сетки

Использование адаптивных свойств CSS, таких как процент и как создавать адаптивный дизайн?

Вопросы для собеседования с Frontend-разработчиком

Вопросы для собеседования с опытным Frontend-разработчиком

43. Что такое JavaScript?

JavaScript - это язык сценариев высокого уровня. Это язык с динамической типизацией. JavaScript используется для добавления динамических элементов и стилей в HTML-документ, чтобы сделать веб-страницу более интерактивной и привлекательной. JavaScript используется как во frontend, так и в backend.

Доступно множество фреймворков и библиотек JavaScript, которые можно использовать во интерфейсе, таких как ReactJS, Vue.JS, AngularJS, NextJS и др. Для серверной разработки доступен только один фреймворк JavaScript, который является Node.JS.

44. В чем разница между == и === в JavaScript?

The == оператор известен как оператор double equal to в JavaScript и === -оператор triple equal to . Оба этих оператора являются операторами сравнения и используются для сравнения значений операндов. Эти операторы практически одинаковы, между ними есть только одно отличие. Оператор == проверяет только значения операндов и возвращает true, если значения совпадают. С другой стороны, оператор === проверяет не только значения операндов, но и типы операндов. Возвращает true, только если значения и тип операндов одинаковы.

3 == "3" // Returns true

3 === "3" // Returns false

45. Как отложить запуск обработчика событий элемента, если он зависит от внешнего скрипта, загрузка которого занимает некоторое время?

Чтобы отложить обработчик события элемента, если это зависит от внешнего скрипта, загрузка которого занимает некоторое время. Вы можете использовать отложить атрибут внутри тега script при добавлении внешнего файла JavaScript. The отложить атрибут подтверждает, что скрипт будет выполнен после анализа HTML.

<script src='external-script-file.js' defer></script>

Теперь вам нужно динамически прикрепить прослушиватель событий к элементу. Чтобы дождаться загрузки файла внешнего скрипта, мы можем прикрепить загрузить или DOMContentLoaded события перед выполнением скрипта и после завершения загрузки вы можете подключить прослушиватель событий к своему элементу, проверив любую функцию или переменную из внешнего скрипта, загружена она или нет, и запустить код относительно нее. Обратитесь к приведенному ниже коду для понимания:

document.addEventListener('load', function(){

if(typeof coder === "boolean"){

const myBtn = document.getElementById('myBtn');

myBtn.addEventListener('click', function(){

console.log("Button is Clicked!!");

})

}

})

В приведенном выше коде мы предполагаем, что файл внешнего скрипта содержит переменную coder имеет логический тип и использует эту переменную для проверки того, загружен внешний скрипт или нет.

46. Оптимальная стратегия для победы в игре, где, скажем, я начинаю с 1, противник может назвать число X в пределах диапазона [2, 11]. Затем я должен назвать число в диапазоне [X + 1, X + 10], затем оппонента, затем себя и так далее. Выигрывает тот, кто в итоге скажет 100, и игра заканчивается.

Оптимальная стратегия для победы в этой игре - поставить вашего оппонента в ситуацию, когда у него нет выбора назвать какое-то число, более близкое к 100, и тогда у вас есть 100 в диапазоне от этого числа + 1 до этого числа + 10, так что вы можете сказать 100 первым и выиграть игру.

47. Почему вы бы использовали прототип в JS?

Прототип - очень важная и фундаментальная концепция JavaScript. Это базовая концепция объектно-ориентированного программирования и цепочки прототипов. Приведенный ниже список включает некоторые причины использования prototype в JavaScript.

Как динамически изменять свойства и методы объекта.

Используется при реализации наследования.

Разрешить совместное использование свойств и методов объекта.

JavaScript - это язык, основанный на прототипах.

Прототипы можно использовать для создания общих методов, позволяющих экономить память.

48. Что означает ‘это’ в JS?

В this ключевое слово в JavaScript - это идентификатор, основанный на области действия функции или переменной, из которой оно вызывается. Поведение этого ключевого слова в зависимости от области, в которой оно используется, меняется. На глобальном уровне это ключевое слово относится к глобальному window объект. В приведенном ниже примере это ключевое слово будет относиться к области видимости объекта, функция которого вызывается:

const myObj = {

name: "GeeksforGeeks",

stats: function(desc){

this.desc = desc;

}

}

console.log(myObj.stats("A Computer Science portal for Geeks."));

49. Пространства имен в JavaScript

Пространства имен в JavaScript - это базовая парадигма программирования, которая обеспечивает инкапсуляцию некоторых переменных функций. Таким образом, если функции и переменные с одинаковыми именами определены, программа в каком-то другом месте не сворачивается и работает должным образом. Вы также можете использовать объекты для достижения того же эффекта.

//Declaring namespace

let myNamespace = {

nameSpace: "Creating a name space in JavaScript"

}

//Accessing namespace identifiers

console.log(myNamespace.nameSpace);

//Output: Creating a name space in JavaScript

50. Разница между null и undefined в JS.

Значение не определено и null в основном это значения, которые переменные или функция получают по умолчанию. Значения обоих этих значений различны, не используйте эти значения взаимозаменяемо.

Неопределенное значение:

Значение undefined - это значение по умолчанию, которое присваивается переменной, которая объявлена, но не инициализирована.

Это также возвращаемое значение функции по умолчанию.

Когда вы пытаетесь получить доступ к некоторому значению или свойству, которого нет в части object , оно возвращает значение undefined .

Нулевое значение:

Его можно присвоить переменной, чтобы сделать ее пустой переменной.

Это означает, что переменная доступа отсутствует в коде.

51. Объясните замыкания в JS на примере инструкций в цикле

Замыкание в JavaScript формируется, когда функция определяется внутри другой функции или возвращается функцией. Способность внутренней функции запоминать свою область действия и область действия цепочки ее родительской функции называется замыканием. Функция закрытия может получить доступ к переменным своей родительской функции, даже если она выполняется.

function sum(a){

return function sum2(b){

return a+b;

}

}

for(let i=1; i<=5; i++){

console.log(sum(i)(i+1));

}

52. Что такое цикл событий в JavaScript?

JavaScript - это синхронный и однопоточный язык, что означает, что JavaScript выполняет код построчно за один раз и не требует ожидания выполнения какого-либо кода. Цикл событий - это механизм, встроенный в движок JavaScript для эффективной обработки асинхронных задач. Цикл событий непрерывно проверяет стек вызовов и очередь обратного вызова. Если он обнаруживает, что стек вызовов пуст, он немедленно переводит асинхронную задачу из очереди обратного вызова в стек вызовов для выполнения.

53. Объясните, как поднять JS?

Hoisting - это концепция в JavaScript, которая позволяет нам получать доступ или извлекать значения функций и переменных еще до инициализации или присвоения им значений. Причиной этого является выделение памяти для переменных и функций на первом этапе выполнения.

console.log(hoistedVariable); // Undefined

var hoistedVariable = 25;

console.log(hoistedVariable); // 25

54. Какие существуют различные типы данных в JS?

JavaScript - это язык с динамической типизацией, что означает, что в отличие от других языков программирования, таких как C, C ++, тип переменных определяется во время выполнения, а не во время компиляции. В JavaScript доступно два типа типов данных:

Примитивный тип данных: Это предопределенные типы данных, такие как строка, число, логическое значение, неопределенное, null, BigInt и т.д.

Непримитивный тип данных: Это типы данных, производные от примитивных типов данных, таких как массивы и Объекты.

55. Если typeof([]) является object, то каково содержимое и длина b в приведенном ниже коде?

let b = [];

b.v = 10;

b.push(11);

Массивы в JavaScript - это особый тип данных, который может содержать числовые значения, а также значения с ключом. В приведенном выше коде,

строка var b = [], создает пустой массив,

затем строка b.v=10, добавляет ключ в массив со значением 10 и

наконец, строка b.push(11), помещает числовое значение 11 в массив.

Содержимое массива выглядит следующим образом: [11, v: 10], и длина массива равна 1, потому что в массиве присутствует только одно числовое значение.

56. Различные способы создания объектов в JS?

В JavaScript мы можем создавать объекты, используя различные синтаксисы. Ниже приведены несколько способов создания объекта в JavaScript:

Использование объектного литерала

let obj = {}

Использование функции построения

function Student(name, class){

this.name = name;

this.class = class;

}

let obj = new Student("Neha", 8);

Использование метода Object.create()

const obj = Object.create(prototype)

Использование класса

class Student{

constructor(name, class){

this.name = name;

this.class = class;

}

}

let obj = new Student("Neha", 8);

Использование конструктора объектов

let obj = new Object();

obj.name = "Neha";

57. Как реализовать наследование в JS?

В JavaScript наследование может быть реализовано двумя различными способами:

Прототипное наследование: его можно реализовать, как показано ниже.

// Parent constructor function

function Student(name) {

this.name = name;

}

// Adding a method to the prototype of the parent

Student.prototype.getName = function() {

console.log('My name is ' + this.name);

};

// Child constructor function

function Rahul(name, std) {

// Call the parent constructor using call or apply

Student.call(this, name);

// Add properties specific to the child

this.std = std;

}

// Inherit from the parent prototype

Rahul.prototype = Object.create(Student.prototype);

// Add a method specific to the child

Rahul.prototype.speak = function() {

console.log('Hello World!');

};

// Create an instance of the child

const myObj = new Rahul('Rahul', '8th');

// Access inherited method from the parent

myObj.getName(); // Output: My name is Rahul

// Access method specific to the child

myObj.speak(); // Output: Hello World!

Наследование на основе классов: Он использует концепции объектно-ориентированного программирования JavaScript и наследует свойства родительского класса с помощью расширяет ключевое слово, как показано ниже.

class Parent{

// Some properties

}

class Child extends Parent{

// Some properties

}

58. Объясните методы call(),apply() и bind() в JavaScript.

Эти методы являются встроенными методами JavaScript и используются для вызова функций с указанным контекстом и this значением контекста. Эти методы также можно использовать для установки значения этого ключевого слова в какой-либо другой контекст при вызове функций.

метод call(): Он используется для вызова функций с заданным это значение идентификатора и отдельные множественные аргументы.

let myObj = {

implementCall: function(name, desc){

this.name = name;

this.desc = desc;

console.log("Implementing Call method");

console.log(this.name, this.desc);

}

}

let func = myObj.implementCall;

func.call(this, "GeeksforGeeks,", "A Computer Science portal for all geeks");

метод apply(): Метод apply() в JavaScript почти такой же, как метод call(), между ними есть только одно отличие, то есть метод apply позволяет передавать функции несколько аргументов вместе в виде массива.

let myObj = {

implementApply: function(name, desc){

console.log("Implementing Apply method");

console.log(name, desc);

}

}

let func = myObj.implementApply;

func.apply(this, ["GeeksforGeeks,", "A Computer Science portal for all geeks"]);

метод bind(): Методы call() и apply() вызывают функцию сразу после приложения, но метод bind() используется для привязки функции к некоторой переменной с переданными параметрами и областью действия это идентификатор, чтобы функцию можно было вызвать позже, когда вам это понадобится.

let myObj = {

implementApply: function(name, desc){

console.log("Implementing Bind method");

console.log(name, desc);

}

}

let bindedFunc = myObj.

implementApply.bind(myObj, "GeeksforGeeks,", "A Computer Science portal for all geeks");

bindedFunc();

59. Что такое делегирование событий в JavaScript?

Делегирование событий - это метод в JavaScript, который используется для повышения производительности и сокращения использования памяти нашего веб-сайта путем прикрепления единственного прослушивателя событий к родительскому элементу в случае, когда вам нужно прикрепить прослушиватели событий к каждому дочернему элементу этого элемента. Вы можете управлять изменениями на веб-сайте, используя только одно событие с родительским элементом.

60. Расскажите мне о ключевых функциях JavaScript.

JavaScript предоставляет множество функций, некоторые из них перечислены ниже:

Это однопоточный язык.

Динамический ввод переменных.

Прототипное и классическое наследование

Первоклассные функции

Функции более высокого порядка.

Подъем, закрытие и т.д.

61. Для чего используется директива “use strict” в JavaScript?

Директива use strict используется для написания чистого кода JavaScript, который менее подвержен ошибкам. Он выявляет распространенные ошибки программирования, такие как присвоение переменной без ее объявления или передача функции разных параметров с одинаковыми именами и т.д.

62. Что такое распространение событий в JavaScript?

Распространение событий определяет поведение, по которому будут распространяться события, когда они будут присоединены к дочерним и родительским элементам. Существует два способа распространения событий.

Бурление событий

Запись событий

63. Как определить "пузырящееся" событие в JavaScript?

Обработка событий - это процесс, при котором событие распространяется от целевого элемента обратно к его предкам в дереве DOM, что означает, что при нажатии на самый внутренний элемент событие переместится к самому внешнему предку или родительскому элементу в дереве DOM. Это поведение по умолчанию при распространении событий.

64. Что такое захват событий?

Захват события прямо противоположен развертыванию события. В данном случае событие распространяется от самого внешнего элемента к самому внутреннему элементу. Это можно включить, передав дополнительный параметр как true в addEventListener() метод на момент прикрепления события.

65. Что такое функции обратного вызова в JavaScript?

Функция обратного вызова в JavaScript - это функция, которая может быть передана в качестве параметра другой функции, а затем эта функция несет ответственность за ее вызов позже в определенное время. Например: addEventListener() метод принимает функцию обратного вызова и вызывает ее при возникновении переданного события, setTimeout() метод также использует функцию обратного вызова, которая будет вызвана позже по истечении пройденного периода времени.

66. Что такое ад обратного вызова и как его избежать?

Ад обратного вызова относится к ситуации, в которой используется несколько вложенных функций обратного вызова, т. Е. функция содержит несколько вложенных функций, и все они вызываются путем передачи им функции обратного вызова в качестве параметра. В основном это происходит, когда вы имеете дело с асинхронными операциями. Обещания и асинхронность / ожидание заявления можно использовать, чтобы избежать ада с обратным вызовом.

67. Объясните обещания на JavaScript.

Обещание - это объект, который представляет завершение или сбой асинхронной задачи и ее результирующее значение в зависимости от состояния. У обещания есть три состояния.:

ожидается: Это начальное состояние, в котором обещание не отклоняется и не выполняется.

выполнено: Это отражает успешное выполнение обещания, при котором задача выполнена.

отклонено: Это означает, что асинхронная задача завершается неудачно.

68. Для чего используется async await в JavaScript?

Оператор async используется для представления асинхронной задачи, для решения которой требуется некоторое время. Оператор await используется для ожидания разрешения или выполнения обещания или асинхронной задачи, чтобы продолжить выполнение асинхронной функции.

69. Объясните концепцию карринга в JavaScript?

Каррирование - это метод, который используется для преобразования функции, которая принимает n параметров, в цепочку из n функций, в которой каждая функция содержит только один параметр.

function normalFunction(a, b){

return a\*b;

}

normalFunction(3, 2) // Output: 6

function curryingFunction(a){

return function(b){

return a\*b;

}

}

curryingFunction(3)(2); // Output: 6

70. Какова цель передачи атрибутов defer или async тегу script?

отложить и асинхронный атрибуты используются для загрузки скрипта определенным образом, как описано ниже:

отложить: Он параллельно извлекает HTML и скрипт, как только извлекается весь скрипт, он прекращает извлекать HTML и сначала выполняет скрипт, затем снова запускает выборку HTML и выполняет его после полной выборки.

<script src="script\_file\_path" defer></script>

асинхронность: Атрибут async позволяет серверу извлекать скрипт и HTML параллельно, и как только он извлекает их оба, он начинает выполнять скрипт и только после этого выполняет HTML.

<script src="script\_file\_path" async></script>

71. Объясните блок try catch в JavaScript.

Блоки try и catch используются для эффективной обработки различных состояний операции. Блок try проверяет написанный внутри него код на наличие ошибок. Если код выполняется с ошибками, то блокируется catch обнаруживает эту ошибку и отображает их пользователю. В противном случае будет выполнен код, написанный внутри блока try для успешного выполнения условия.

try{

// try block code

}

catch(error){

// Show errors to users

}

72. Как остановить распространение событий в JavaScript?

Есть встроенный метод например,stopPropagation() предоставлены событиями JavaScript, которые можно использовать для остановки распространения событий. Метод может быть реализован следующим образом:

element.addEventListener('event', function(e){

e.stopPropagation();

})

73. Как предотвратить поведение события по умолчанию в JavaScript?

The e.preventDefault() это встроенный метод, предоставляемый JavaScript, который можно использовать для предотвращения поведения события JavaScript по умолчанию.

74. Опишите концепцию CORS?

CORS расшифровывается как Совместное использование ресурсов разных источников. Это метод, используемый браузерами для повышения безопасности нашей веб-страницы. Веб-браузеры используют эту функцию для предотвращения запросов от одного домена к другому домену.

75. Объясните, как происходит дебатирование в JavaScript?

Это метод, используемый для повышения производительности веб-страницы за счет того, что задачи, отнимающие много времени, не так часто запускаются в коде. Один из вариантов использования деблокирования - справиться с ситуацией, когда при каждом нажатии клавиши выполняются частые запросы к серверу. Например: функция поиска на разных веб-страницах.