1. Как вы можете описать процесс распространения событий в JavaScript (Event Propagation)? Расскажите о его основных фазах. Какие есть способы применения знаний о фазах распространения событий на практике?

Ответ: К примеру у нас есть кнопка на сайте и мы по ней нажимаем. Происходит событие (клик) и он движется по структуре нашего ДОМ дерева, HTML. Существуют три фазы. 1- Захват, это когда наше событие движется с верхнего элемента до нужного целевого элемента сверху вниз. 2- Целевая фаза, это когда событие дошло до того элемента на который нажали. 3- Фаза всплытия, когда событие обратно поднимается от целевого элемента. Применял обычно для остановки всплытия через метод stopPropagation

().

1. Расскажите, что такое Promise в JavaScript, как он работает, какие

основные состояния у него бывают? Каким еще способом можно обработать асинхронный код? Какую роль в асинхронности играет событийный цикл (Event Loop)?

Ответ: Promise это объект который дает возможность обрабатывать асинхронные операции. У него есть три состояния. Ожидание, это у нас самое начальное состояние. Выполнено, это когда наша операция завершилась успешно. Отклонено, это когда операция завершилась с какой то ошибкой. Для того чтобы обработать состояния используются методы then()-для успешного завершения и метод catch() -для обработки ошибок. А так же есть метод finally() он выполнится в любом случае. С Event Loop мы можем выполнять много операций параллельно, при этом не блокируем основной поток. Операции будут выполнены последовательно.

1. Что такое ООП, и какие его ключевые принципы вы могли бы описать? При объяснении приводите примеры. Максимально подробно расскажите о том, как ООП реализовано в JavaScript.

Ответ: ООП- это подход к программированию, где всё представляется в виде объектов. Взять к примеру автомобиль. Класс- это у нас будет схема или чертеж самого автомобиля. Объект – это сделанный автомобиль по схеме. Свойства-это комплектация автомобиля. Методы – это любые механические или электронные действия, включить свет, включить печку и т.д. Есть основные принципы. Это инкапсуляция- это когда мы скрываем внутренние методы объекта от прямого доступа извне. Есть примеру приватные поля и для того чтобы с ними работать мы создадим отдельный метод, через который мы можем уже с этим полем что то делать. Наследование – это когда мы создаем классы таким образом, что один класс может получать атрибуты и методы другого класса. Тем самым позволяет не повторяться, а переиспользовать код повторно. Полиморфизм - это когда мы можем использовать один и тот же метод для разных объектов, но при этом каждый объект будет реализовывать этот метод по своему. К примеру у нас есть два объекта и там у нас реализованы одинаковые методы но с разными подключениями к базе данных. Мы можем сделать перебор объектов и в зависимости какое нам нужно подключение то и тот объект будем обрабатывать. Абстракция - это когда мы пытаемся важные свойства объекта выделить и скрыть какие то не важные делали реализации от пользователя. К примеру автомобиль, есть педаль и для пользователя скрыта вся логика работы электроники. ООП в JS реализовано на основе прототипного наследования. У каждого объекта имеется ссылка на прототип. К примеру мы создаем конструктор класса и потом через его наименование мы с помощью метода prototype можем присвоить ему к примеру какой нибудь метод. Так же есть наследование классов. Приватные поля которые начинаются с #. Напрямую мы ими не может управлять.

1. Опишите максимально детально, как браузер обрабатывает URL,

начиная с момента его ввода в адресную строку и заканчивая

отображением страницы. Какие технологии позволяют ускорить

процесс? Какие проблемы (безопастность, междоменное

взаимодействие) могут возникать в процессе?

Ответ: Сам URL у нас разбит на части. Протокол, домен, порт, путь, параметры. Когда мы ввели url то браузер проверяет наличие кэша по этому адресу, так как может уже быть сохраненные IP адреса нашему домену. Если нет в кэше ничего то браузер обращается к DNS с запросом найти IP для нашего хоста. Полученный ip адрес сохраняется в кэше DNS и отправляется обратно браузеру. Дальше мы устанавливаем TCP соединение с IP адресов сервера. После установки соединения браузер отправляет HTTP запрос серверу, запрашивает нужные нам данные. К примеру нам приходит HTML документ. Браузер парсит HTML документ и при этом он получает дальше нужные нам файлы, такие как css, JS файлы, картинки и т.д. Происходит рендеринг страницы, строит ДОМ дерево. Применяет к нему стили. Технологие которые могут ускорить процесс: Это кэширование, так же можно сделать предзагрузку ресурсов, оптимизация изображений, ленивая загрузка. Злонамеренный код может внедрятся на страницу и выполнятся в браузере. Поделка межсайтовых запросов. Могут так же перехватывать данные между клиентом и сервером. Cors ограничения, могут блокироваться запросы. Куки могут не отправляться  в cross-origin запросах.