Веб-производительность

Веб-производительность — это объективные измерения и восприятие пользователем времени загрузки и времени выполнения. Веб-производительность — это то, сколько времени требуется сайту, чтобы загрузиться, стать интерактивным и отзывчивым, а также насколько плавно отображается контент во время взаимодействия с пользователем — плавна ли прокрутка? кнопки кликабельны? Быстро ли загружаются и отображаются всплывающие окна, и плавно ли они анимируются при этом? Веб-производительность включает в себя как объективные измерения, такие как время загрузки, количество кадров в секунду и время перехода в интерактивный режим, так и субъективные ощущения того, сколько времени по ощущениям потребовалось для загрузки контента.

Чем дольше сайт отвечает, тем больше пользователей покидают сайт. Важно свести к минимуму время загрузки и отклика, а также добавить дополнительные функции, чтобы скрыть задержку, сделав опыт как можно более доступным и интерактивным как можно скорее, при этом асинхронно загружая более длинные хвостовые части опыта.

Существуют инструменты, API и рекомендации, которые помогают нам измерять и повышать производительность сети. Мы рассмотрим их в этом разделе:

Основные руководства по производительности Производительность анимации и частота кадров Анимация в Интернете может быть выполнена с помощью SVGAnimationElement, window.requestAnimationFrame, в том числе canvasи WebGL_API, CSS animation, video, анимированных GIF и даже анимированных PNG и других типов изображений. Стоимость анимации свойства CSS может варьироваться от одного свойства к другому, а анимация дорогостоящих свойств CSS может привести к рывкам, поскольку браузер изо всех сил пытается достичь плавного FPS.

Критический путь рендеринга Критический путь рендеринга — это последовательность шагов, которые браузер выполняет для преобразования HTML, CSS и JavaScript в пиксели на экране. Оптимизация критического пути рендеринга повышает производительность рендеринга. Критический путь рендеринга включает объектную модель документа (DOM), объектную модель CSS (CSSOM), дерево рендеринга и макет.

Производительность анимации CSS и JavaScript Анимация имеет решающее значение для приятного взаимодействия с пользователем во многих приложениях. Существует множество способов реализации веб-анимации, например анимации на основе CSS transition/ animationJavaScript (с использованием Window.requestAnimationFrame). В этой статье мы анализируем разницу в производительности между анимацией на основе CSS и на основе JavaScript.

Ленивая загрузка Ленивая загрузка — это стратегия определения ресурсов как неблокирующих (некритических) и загрузки их только при необходимости. Это способ сократить длину критического пути отрисовки , что приводит к сокращению времени загрузки страницы.

Время навигации и ресурсов Время навигации — это метрика, измеряющая события навигации по документу в браузере. Тайминги ресурсов — это подробные измерения сетевых таймингов, касающиеся загрузки ресурсов приложения. Оба предоставляют одни и те же свойства только для чтения, но время навигации измеряет время основного документа, тогда как время

ресурса предоставляет время для всех активов или ресурсов, вызываемых этим основным документом, и ресурсов, запрошенных ресурсами.

Оптимизация производительности запуска Улучшение производительности при запуске часто является одной из самых важных оптимизаций производительности, которую можно сделать. Сколько времени занимает запуск вашего приложения? Блокирует ли это устройство или браузер пользователя во время загрузки приложения? Это заставляет пользователей беспокоиться о том, что ваше приложение потерпело крах или что-то еще не так. Хороший пользовательский опыт включает в себя быструю загрузку вашего приложения. В этой статье приведены советы и предложения по повышению производительности как для написания новых приложений, так и для переноса приложений в Интернет с других платформ.

Бюджеты производительности Бюджет производительности — это предел для предотвращения регрессии. Он может применяться к файлу, типу файла, всем файлам, загруженным на страницу, определенной метрике (например, времени до интерактивности), пользовательской метрике (например, времени до главного элемента) или пороговому значению за период времени.

Основы производительности Производительность означает эффективность. В контексте открытых веб-приложений в этом документе в общих чертах объясняется, что такое производительность, как платформа браузера помогает ее повысить, а также какие инструменты и процессы можно использовать для ее тестирования и улучшения.

Мониторинг производительности: RUM против синтетического мониторинга Синтетический мониторинг и мониторинг реальных пользователей (RUM) — это два подхода к мониторингу и анализу производительности сети. RUM и синтетический мониторинг обеспечивают разные взгляды на производительность и имеют преимущества, хорошие варианты использования и недостатки. RUM, как правило, лучше всего подходит для понимания долгосрочных тенденций, тогда как синтетический мониторинг очень хорошо подходит для регрессионного тестирования и устранения краткосрочных проблем с производительностью во время разработки. В этой статье мы определяем и сравниваем эти два подхода к мониторингу производительности.

Наполнение страницы: как работают браузеры Пользователям нужен веб-интерфейс с контентом, который быстро загружается и с которым легко взаимодействовать. Поэтому разработчик должен стремиться к достижению этих двух целей.

Рекомендуемые сроки веб-производительности: сколько времени слишком долго? Нет четких установленных правил относительно того, что представляет собой медленный темп при загрузке страниц, но есть конкретные рекомендации по указанию загрузки контента (1 секунда), бездействия (50 мс), анимации (16,7 мс) и реакции на ввод пользователя (от 50 до 50 мс). 200 мс).

Понимание задержки Задержка — это время, необходимое пакету данных для перемещения от источника к месту назначения. С точки зрения оптимизации производительности важно оптимизировать, чтобы уменьшить причины задержки, и протестировать производительность сайта, имитируя высокую задержку, чтобы оптимизировать для пользователей с плохим подключением. В этой статье объясняется, что такое задержка, как она влияет на производительность, как измерить задержку и как ее уменьшить.

Использование предварительной выборки DNS DNS-prefetch— это попытка разрешить

доменные имена до того, как будут запрошены ресурсы. Это может быть файл, загруженный позже, или цель ссылки, по которой пользователь пытается перейти.

Учебники для начинающих Область обучения веб-производительности MDN содержит современные актуальные учебные пособия, посвященные основам производительности. Начните здесь, если вы новичок в производительности:

Веб-производительность: краткий обзор Обзор схемы обучения веб-производительности. Начните свое путешествие здесь.

Что такое веб-производительность? В этой статье модуль начинается с хорошего рассмотрения того, что такое производительность на самом деле — сюда входят инструменты, метрики, API, сети и группы людей, которых нам нужно учитывать, когда мы думаем о производительности, и как мы можем сделать производительность частью нашей сети. рабочий процесс разработки.

Как пользователи воспринимают производительность? Более важно, чем скорость вашего веб-сайта в миллисекундах, это то, насколько быстро ваши пользователи воспринимают ваш сайт. На это восприятие влияет фактическое время загрузки страницы, бездействие, реакция на взаимодействие с пользователем, а также плавность прокрутки и других анимаций. В этой статье мы обсудим различные метрики загрузки, анимацию и показатели отклика, а также лучшие практики для улучшения восприятия пользователем, если не фактическое время.

Основы веб-производительности В дополнение к внешним компонентам HTML, CSS, JavaScript и мультимедийным файлам существуют функции, которые могут замедлить работу приложений, и функции, которые могут сделать приложения субъективно и объективно быстрее. Существует множество API, инструментов разработчика, лучших и плохих практик, связанных с веб-производительностью. Здесь мы представим многие из этих функций на базовом уровне и предоставим ссылки на более подробные сведения для повышения производительности по каждой теме.

Особенности производительности HTML Некоторые атрибуты и исходный порядок вашей разметки могут повлиять на производительность вашего веб-сайта. Сведя к минимуму количество узлов DOM, обеспечив использование наилучшего порядка и атрибутов для включения такого контента, как стили, сценарии, мультимедиа и сторонние сценарии, вы можете значительно улучшить взаимодействие с пользователем. В этой статье подробно рассматривается, как можно использовать HTML для обеспечения максимальной производительности.

Мультимедиа: изображения и видео Наиболее дешевым плодом веб-производительности часто является медиа-оптимизация. Возможно обслуживание различных медиафайлов в зависимости от возможностей, размера и плотности пикселей каждого пользовательского агента. Дополнительные советы, такие как удаление звуковых дорожек из фоновых видео, могут еще больше повысить производительность. В этой статье мы обсудим влияние контента видео, аудио и изображений на производительность, а также способы обеспечения минимального влияния.

Особенности производительности CSS CSS может быть менее важным направлением оптимизации для повышения производительности, но есть некоторые функции CSS, которые влияют на производительность больше, чем другие. В этой статье мы рассмотрим некоторые свойства CSS, влияющие на производительность, и предлагаем способы обработки стилей, чтобы гарантировать отсутствие негативного влияния на производительность.

Рекомендации по производительности JavaScript JavaScript при правильном использовании может обеспечить интерактивный и захватывающий веб-опыт — или он может значительно снизить время загрузки, время рендеринга, производительность в приложении, время автономной работы и пользовательский опыт. В этой статье описаны некоторые передовые методы работы с JavaScript, которые следует учитывать, чтобы обеспечить максимально возможную производительность даже сложного контента.

Мобильная производительность Поскольку веб-доступ на мобильных устройствах так популярен, а все мобильные платформы имеют полноценные веб-браузеры, но, возможно, имеют ограниченную пропускную способность, ЦП и время автономной работы, важно учитывать производительность вашего веб-контента на этих платформах. В этой статье рассматриваются вопросы производительности для мобильных устройств.

Использование API производительности API производительности В этом руководстве описывается, как использовать Performanceинтерфейсы, определенные в стандарте High-Resolution Time .

API синхронизации ресурсов Загрузка ресурсов и синхронизация загрузки этих ресурсов, включая управление буфером ресурсов и работу с CORS.

График производительности Стандарт Performance Timeline определяет расширения интерфейса Performanceдля поддержки измерения задержки на стороне клиента в приложениях. Вместе эти интерфейсы можно использовать для выявления узких мест производительности приложения.

API пользовательского времени Создавайте временные метки для конкретных приложений, используя типы записей «mark» и «measure» пользовательского API синхронизации , которые являются частью временной шкалы производительности браузера.

API маяка Интерфейс Beacon планирует асинхронный и неблокирующий запрос к веб-серверу.

API-интерфейс наблюдателя за пересечением Научитесь определять время видимости элементов с помощью API Intersection Observer и получать асинхронные уведомления, когда интересующие элементы становятся видимыми.

Другая документация Функции производительности Firefox Profiler Этот веб-сайт предоставляет информацию о том, как использовать и понимать функции производительности в ваших инструментах разработчика, включая дерево вызовов , Flame Graph , Stack Chart , Marker Chart и Network Chart .

Профилирование с помощью встроенного профилировщика Узнайте, как профилировать производительность приложения с помощью встроенного профилировщика Firefox.

Глоссарий терминов Маяк сжатие Бротли Подсказки для клиентов Разделение кода CSSOM Шардинг домена Эффективный тип соединения Первая содержательная краска Первый процессор простаивает Задержка первого ввода Первый интерактив Первая осмысленная краска Первая краска HTTP HTTP/2 Джанк Задержка Ленивая загрузка Длинная задача Сжатие без потерь Сжатие с потерями Основная нить Минификация Дросселирование сети Пакет Время загрузки страницы Предсказание страницы Разобрать Воспринимаемая производительность Предварительная выборка Пререндеринг QUIC ЖЕЛЕЗНОДОРОЖНЫЙ Мониторинг реального пользователя Сроки ресурсов Время приема-передачи (RTT) Время

сервера Спекулятивный анализ Индекс скорости SSL Синтетический мониторинг TCP-рукопожатие Медленный запуск TCP Время до первого байта Время интерактива TLS Протокол управления передачей (TCP) Сотрясение дерева Веб-производительность