Engineer

Service worker

скрипт, що відповідає на запити без мережі

Що це таке

Service worker — це скрипт, який сайт реєструє в браузері і який браузер далі виконує окремо від будь-якої сторінки; мережеві запити сторінок, якими він керує, проходять через нього. Service workers — різновид web worker: кожен працює у власному потоці, без документа й без вікна. Браузер запускає його, коли є що обробити, і може зупинити, щойно роботи не лишилося, тож усе, що він тримає в змінній, до наступної події може зникнути.

Сторінка встановлює його одним викликом, і специфікація ставить до цього три умови. Сторінка мусить бути безпечним контекстом, що на практиці означає HTTPS, а для розробки дозволено `localhost` та адреси loopback. Скрипт мусить мати те саме походження (origin), що й сторінка, тож лежати на CDN він не може, хоча імпортувати звідти код йому дозволено. І реєстрація охоплює певну область, scope: усталено це тека, у якій лежить скрипт, і нічого ширшого, хіба що сервер надішле разом зі скриптом заголовок `Service-Worker-Allowed`.

```
// на сторінці
navigator.serviceWorker.register("/sw.js");

// у /sw.js
async function answer(request) {
    const copy = await caches.match(request);
    return copy ?? fetch(request);
}

self.addEventListener("fetch", (event) => {
    event.respondWith(answer(event.request));
});
```

Такий service worker відповідає на кожен запит власною копією, якщо вона в нього є, а якщо ні — даними з мережі. Покласти туди копії — завдання життєвого циклу. Щойно скрипт завантажено, браузер надсилає подію `install`, слушну мить, щоб завантажити й зберегти все потрібне; якщо проміс, переданий у `event.waitUntil()`, буде відхилено, усе встановлення зазнає невдачі. Потім настає `activate`, час прибрати те, що лишила старша версія, і лише після цього service worker отримує події `fetch`. Сторінку прив'язують до її service worker під час завантаження, тож сторінка, яка зареєструвала найперший, працює без нього, доки її не перезавантажать, хіба що він викличе `clients.claim()`.

Копії зберігаються в Cache Storage, яке визначає та сама специфікація: це іменовані кеші з пар запитів і відповідей, які service worker сам наповнює, читає й видаляє. Вони відокремлені від HTTP-кешу браузера. Запис не має терміну дії, сам ніколи не оновлюється й переживає оновлення скрипту, тож лишається, доки його не видалить сам service worker або браузер не очистить сховище сайту.

Застосуйте поняття до рішення

Застосуйте «Що таке service worker?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Інтерфейси, що швидко завантажуються, працюють на телефоні й поводяться однаково в усіх браузерах ваших клієнтів.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Керувала full-stack-розробкою GIS-карт — full-stack-картографія — від бази даних до браузера

Керувала full-stack-розробкою GIS-карт Зв'язатися з нами →

Чому це важливо

Service workers дають сайту змогу явно керувати своєю роботою зовсім без мережі. Service worker вирішує для кожного запиту окремо, у коді, який написав сам сайт, звідки брати відповідь: із мережі, із власних копій чи з обох, і саме це робить встановлений вебзастосунок придатним до роботи на кораблі чи в підвалі. До нього також доходять push-повідомлення, коли жодної сторінки сайту не відкрито.

Більшість таких скриптів зводиться до вибору стратегії для кожного виду запитів, і Workbox, бібліотека для їх написання, називає п'ять. «Спершу кеш» відповідає з копії й іде в мережу, лише коли копії немає, і пасує файлам, які ніколи не змінюються під тією самою назвою. «Спершу мережа» питає

мережу й повертається до копії, коли мережа підводить, і пасує сторінкам. Stale-while-revalidate відповідає з копії та оновлює її у фоні. «Лише мережа» і «лише кеш» — дві крайності.

Водночас це код, що стоїть між кожною сторінкою свого scope і мережею, на пристрої читача, і він переживає відвідування: помилка в ньому лишається, доки на зміну не прийде виправлений скрипт. Специфікацію збудовано навколо цього ризику. Кожна навігація змушує браузер перевірити, чи є новий скрипт, а інші події роблять це, коли від останньої перевірки минула доба. Усталено основний скрипт браузер звіряє із сервером, а не бере з HTTP-кешу, тож довгий `max-age` на файлі скрипту не втримає зламану версію; імпортовані скрипти перевіряють так само, коли минула доба, або завжди, коли `updateViaCache` має значення `none`. Браузер порівнює завантажений скрипт зі встановленим байт за байтом, і лише різниця запускає оновлення. Shift-reload оминає service worker для цієї однієї навігації, і через це помилка на машині розробника може здаватися виправленою.

Правило очікування — друга половина цього задуму. Нова версія встановлюється поруч зі старою і потім чекає, невикористана, доки жодна відкрита сторінка більше не залежить від старої, тож дві версії ніколи не обслуговують одну сторінку на двох. Виклик `skipWaiting()` дає їй змогу перейняти керування відразу, ціною того, що новий код працюватиме під сторінками, які завантажила стара.

Сховище — остання ціна. Cache Storage займає частину місця, яке сайт має на пристрої читача, а за Storage Standard це місце усталено працює в режимі best-effort: браузер може очистити його, коли на пристрої бракує місця, якщо тільки сайт не попросив постійного сховища й не отримав його.

Як це виглядає на практиці

Потреба в ньому найпомітніша там, де мережа найслабша. Прогресивний вебзастосунок для моряків і польового персоналу обслуговує людей, які часто офлайн або на поганому з’єднанні. Його service worker налаштовано через плагін next-pwa, а кешування Workbox під час виконання застосовує «спершу кеш» до того, що не змінюється від запиту до запиту: шрифтів, зображень, аудіо, відео й ресурсів із CDN, поряд із заголовками HTTP cache-control. Оскільки service workers по-справжньому працюють лише через

HTTPS, поведінку офлайн і встановлення перевіряли через HTTPS на справжніх пристроях, а повторні завантаження надходять із кешу, а не з мережі.

Цей сайт робить протилежний вибір і прямо про це каже. Сторінка про файли cookie повідомляє, що на сайті немає ні service worker, ні IndexedDB, ні Cache Storage: ніщо із сайту не працює у фоні браузера до його відкриття чи після закриття. Сторінки не постачають виконуваного JavaScript. Код пошуку збережено, але вимкнено, і він не реєструє worker.

Сторінка встановлення робить із цього висновок для всіх, хто додає сайт на домашній екран: встановлення дає сайту власне вікно, але не створює керованої офлайн-копії. Цей сайт не постачає service worker чи кероване скриптами сховище для роботи офлайн, тож встановленому застосунку потрібне з'єднання для звичайного використання.

Worker реєструється з виконуваного скрипту, а поточна Content-Security-Policy цього сайту забороняє це через `script-src 'none'` на кожній сторінці. JSON-LD залишається невиконуваними даними. Колишній коментар у шаблоні head описував збірку, де 550 із 793 документів мали `'none'`, а 243 — `'self'`, відповідно до сторінок, що тоді містили фільтр списку. Ці числа стосуються попередньої збірки зі скриптом, а не поточної політики.

Де виникають проблеми

Після розгортання й далі працює стара версія. Новий service worker встановлюється й чекає, поки хоч одна відкрита вкладка ще користується старим, тож читач, який ніколи не закриває ту вкладку, і далі виконує старий код. Викликайте `skipWaiting()`, коли нова версія безпечна під сторінками, які завантажила стара, або покажіть повідомлення про готове оновлення й нехай сторінка попросить service worker, що чекає, викликати цей метод.

Сторінки тижнями лишаються в старій версії. Стратегія «спершу кеш», застосована до HTML, відповідає на кожну навігацію з Cache Storage, записи якого не мають терміну дії й не зважають на заголовки кешування від сервера. Лишайте «спершу кеш» для файлів, назви яких змінюються разом із вмістом, а для сторінок беріть «спершу мережа».

Встановлення зривається, і лишається старий service worker. Одна адреса зі списку кроку встановлення відповідає помилкою, тож `cache.addAll()` відхиляє свій проміс, проміс, переданий у `waitUntil()`, відхиляється разом із ним, а нову версію відкидають. Складайте список із файлів, які справді існують, а необов'язкові файли завантажуйте пізніше, а не під час встановлення.

Service worker оминає більшу частину сайту. Скрипт, що віддається з `/js/sw.js`, має `/js/` і як усталений, і як найширший scope, тож сторінки поза цією текою завантажуються без нього. Віддавайте скрипт із кореня сайту або надсилайте разом із ним `Service-Worker-Allowed: /`.

Сховище на пристрої росте з кожним випуском. Кеші переживають оновлення скрипту, тож кожна версія, яка відкриває кеш під новою назвою, лишає попередній. Додавайте версію до назви кожного кешу, а в обробнику `activate` видаляйте всі кеші, якими поточна версія не користується.

Що почитати далі

- Service Workers — реєстрація, scope, перевірка оновлень, життєвий цикл і Cache Storage.
- Secure Contexts — які походження вважаються потенційно надійними, зокрема `localhost` та адреси loopback.
- Fetch Standard — об'єкти `Request` і `Response`, які обробник `fetch` отримує й повертає, і те, як запит передається до service worker.
- Storage Standard — сховища (storage buckets), у яких живе Cache Storage, оцінки квоти та те, як сайт просить постійного сховища.
- Workbox: workbox-strategies — п'ять стратегій кешування і те, яким запитам пасує кожна.

Словник

Обирайте за фактичними умовами

Опишіть робочий процес, наявні інтерфейси та користувачів, яким потрібна зміна. Ми можемо визначити перевірюваний результат і спосіб передачі, який дасть вашій команді змогу підтримувати його.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке service worker?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/service-worker/