Engineer

Service worker

ett skript som svarar på begäranden när nätverket inte kan

Vad det är

En service worker, i plural service workers, är ett skript som en webbplats registrerar hos webbläsaren. Webbläsaren kör det skilt från alla sidor och låter nätverksförfrågningarna från de sidor det styr passera genom det. Service workers är en sorts web worker: var och en körs i en egen tråd, utan dokument och utan fönster. Webbläsaren startar en när det finns något för den att hantera och kan stoppa den så snart ingenting återstår, så allt den håller i en variabel kan vara borta vid nästa händelse.

En sida installerar en med ett enda anrop, och specifikationen ställer tre villkor för det. Sidan måste vara en säker kontext, vilket i praktiken betyder HTTPS, med `localhost` och loopback-adresserna tillåtna under utveckling. Skriptet måste komma från sidans eget ursprung, så det kan inte ligga på ett CDN, även om det får importera kod därifrån. Och registreringen gäller ett omfång, på engelska scope: som standard mappen där skriptet ligger, och inget bredare om inte servern skickar huvudfältet `Service-Worker-Allowed` tillsammans med skriptet.

```
// på sidan
navigator.serviceWorker.register("/sw.js");

// i /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));
});
```

Den workern besvarar varje förfrågan med sin egen kopia när den har en och från nätverket annars. Att lägga kopiorna där är livscykelns uppgift. När skriptet har hämtats utlöser webbläsaren `install`, tillfället att ladda ned och lagra det som workern kommer att behöva; om ett promise som har lämnats till `event.waitUntil()` där avvisas, misslyckas installationen. Sedan utlöses `activate`, tillfället att städa bort det som en äldre version lämnade efter sig, och först därefter tar workern emot `fetch`-händelser. En sida knyts till sin worker när den laddas, så sidan som registrerade den första workern fortsätter utan någon tills den laddas om, om inte workern anropar `clients.claim()`.

Kopiorna ligger i Cache Storage, som samma specifikation definierar: namngivna cachar med par av förfrågan och svar som workern själv fyller, läser och raderar. De är skilda från webbläsarens HTTP-cache. En post går aldrig ut, uppdateras aldrig av sig själv och överlever en uppdatering av worker-skriptet, så den ligger kvar tills workern raderar den eller webbläsaren rensar webbplatsens lagring.

Använd begreppet i ett beslut

Använd ”Vad är en service worker?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Gränssnitt som laddar snabbt, fungerar i mobilen och beter sig likadant i alla webbläsare dina kunder råkar använda.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Styrde full stack-utvecklingen av GIS-kartor — kartutveckling i full stack, från databas till webbläsare

Styrde full stack-utvecklingen av GIS-kartor Hör av er →

Varför det spelar roll

Service workers ger en webbplats uttrycklig kontroll över hur den fortsätter att fungera helt utan nätverk. En service worker avgör förfrågan för förfrågan, i kod som webbplatsen själv har skrivit, om svaret ska komma från nätverket, från de egna kopiorna eller från båda, och det är det som gör en installerad webbapp användbar på ett fartyg eller i en källare. Push-meddelanden når den också när ingen av webbplatsens sidor är öppen.

De flesta workers handlar i grunden om att välja en strategi för varje sorts förfrågan, och Workbox, ett bibliotek för att skriva dem, namnger fem. CacheFirst svarar från kopian och går till nätverket bara när kopia saknas, vilket passar filer som aldrig ändras under samma namn. NetworkFirst frågar nätverket

och faller tillbaka på kopian, vilket passar sidor. StaleWhileRevalidate svarar från kopian och uppdaterar den i bakgrunden. NetworkOnly och CacheOnly är de två ytterligheterna.

En service worker är också kod som står mellan nätverket och varje sida inom sitt omfång, på läsarens enhet, och den lever längre än besöket: ett misstag i den ligger kvar tills ett rättat skript ersätter den. Specifikationen är byggd kring den risken. Varje navigering får webbläsaren att leta efter ett nytt skript, och andra händelser gör det när ett dygn har gått sedan den senaste kontrollen. Som standard omvalideras huvudskriptet mot servern i stället för att hämtas ur HTTP-cachen, så en lång `max-age` på worker-filen kan inte hålla kvar en trasig version; importerade skript behandlas likadant när ett dygn har gått, eller alltid när `updateViaCache` är `none`. Webbläsaren jämför det hämtade skriptet med det installerade byte för byte, och bara en skillnad startar en uppdatering. En omladdning med Skift hoppar över workern för just den navigeringen, och därför kan en bugg se rättad ut på en utvecklares maskin.

Väntregeln är den andra halvan av den designen. En ny version installeras bredvid den gamla och väntar sedan, oanvänd, tills ingen öppen sida längre är beroende av den gamla, så två versioner delar aldrig på att betjäna samma sida. Ett anrop till `skipWaiting()` låter den ta över direkt, till priset av att ny kod körs under sidor som den gamla laddade.

Lagringen är den sista kostnaden. Cache Storage räknas mot det utrymme som webbplatsen har på läsarens enhet, och enligt Storage Standard är det utrymmet som standard utan garanti, på engelska best-effort: webbläsaren får rensa det när enheten får ont om plats, om inte webbplatsen har bett om beständig lagring och fått den beviljad.

Så ser det ut i praktiken

Skälet att ha en är tydligast där nätverket är som svagast. Den progressiva webbappen för sjöfolk och fältpersonal betjänar människor som ofta är offline eller har en dålig uppkoppling. Dess service worker sätts upp via pluginet next-pwa, och Workbox cachar vid körning med CacheFirst för det som inte ändras från en förfrågan till nästa, typsnitt, bilder, ljud, video och CDN-resurser, tillsammans med HTTP-huvuden för cachning. Eftersom service workers egentligen bara fungerar som de ska över HTTPS testades beteendet offline och vid installation över HTTPS på riktiga enheter, och upprepade laddningar kommer tillbaka från cachen i stället för från nätverket.

Den här webbplatsen gör det motsatta valet och säger det öppet. Sidan om kakor anger att webbplatsen inte har någon service worker, IndexedDB eller Cache Storage: inget från den körs i webbläsarens bakgrund innan läsaren öppnar den eller efter att den stängs. Sidorna levererar ingen körbar JavaScript. Sökkoden finns kvar men är avstängd och registrerar ingen worker.

Installationssidan drar slutsatsen för den som lägger till webbplatsen på en hemskärm: installationen ger den ett eget fönster utan att skapa en hanterad offlinekopia. Den här webbplatsen levererar varken en service worker eller skriptstyrd lagring för offlinebruk, så den installerade appen behöver en anslutning vid normal användning.

En worker registreras från ett körbart skript, och webbplatsens nuvarande Content-Security-Policy förbjuder det med `script-src 'none'` på varje sida. JSON-LD är data och körs inte. En tidigare kommentar i head-mallen beskrev ett bygge där 550 av 793 dokument angav `'none'` och 243 angav `'self'`, motsvarande sidorna som hade listfiltret då. Siffrorna beskriver det tidigare bygget med skript, inte den nuvarande policyn.

Var det går fel

Den gamla versionen fortsätter att svara efter en driftsättning. Den nya workern installeras och väntar sedan så länge någon öppen flik fortfarande använder den gamla, så en läsare som aldrig stänger den fliken fortsätter att köra den gamla koden. Anropa `skipWaiting()` när den nya versionen är säker under sidor som den gamla laddade, eller visa ett meddelande om uppdateringen och låt sidan be den väntande workern att anropa den.

Sidor står kvar på en gammal version i veckor. En CacheFirst-strategi som tillämpas på HTML besvarar varje navigering ur Cache Storage, vars poster aldrig går ut och inte bryr sig om serverns cache-huvuden. Behåll CacheFirst för filer vars namn ändras när innehållet gör det, och använd NetworkFirst för sidor.

En enda trasig adress fäller hela installationen. En adress i installationsstegets lista svarar med ett fel, så `cache.addAll()` avvisas, det promise som lämnades till `waitUntil()` avvisas med det, och den nya versionen kastas medan den gamla workern blir kvar. Bygg listan av filer som finns, och hämta valfria filer senare i stället för vid installationen.

Workern når bara en enda mapp. Ett skript som serveras från `/js/sw.js` har `/js/` som både standardomfång och bredaste möjliga omfång, så sidor utanför den mappen laddas utan det. Servera skriptet från webbplatsens rot, eller skicka `Service-Worker-Allowed: /` med det.

Lagringen på enheten växer med varje release. Cachar överlever en uppdatering av worker-skriptet, så varje version som öppnar en cache under ett nytt namn lämnar den föregående efter sig. Lägg in en version i varje cachenamn, och radera i `activate`-hanteraren varje cache som den nuvarande versionen inte använder.

Läs mer

- Service Workers — registrering, omfång, uppdateringskontrollen, livscykeln och Cache Storage.
- Secure Contexts — vilka ursprung som räknas som potentiellt pålitliga, `localhost` och loopback-adresserna inräknade.
- Fetch Standard — objekten `Request` och `Response` som en `fetch`-hanterare tar emot och returnerar, och hur en förfrågan lämnas över till en service worker.
- Storage Standard — lagringsutrymmena som Cache Storage ligger i, uppskattningar av kvoter och hur en webbplats ber om beständig lagring.
- Workbox: workbox-strategies — de fem cachningsstrategierna och vilken sorts förfrågan var och en passar.

Ordlista

Välj utifrån era faktiska förutsättningar

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är en service worker?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

welcome@engineer.company +45 50 10 61 84 Köpenhamn, Danmark

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/service-worker/