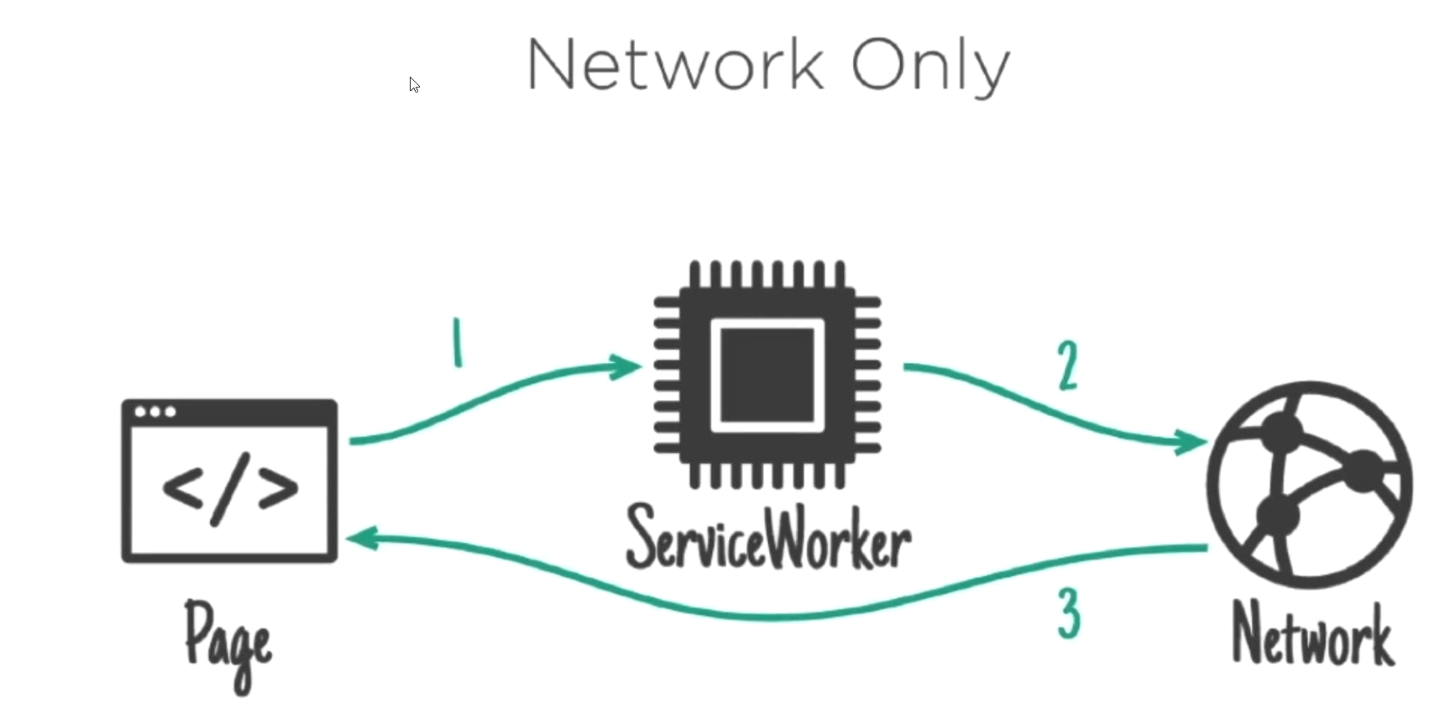
**جلسه یازدهم : WorkBox Dynamic Cache**

نحوه کش کردن محتویات داینامیک : برای اینکه بتوانیم محتویات خود را کش کنیم پنچ روش پیش رو داریم :

1. Network First
2. Cache First
3. Stale While Revalidate
4. Network Only
5. Cache Only

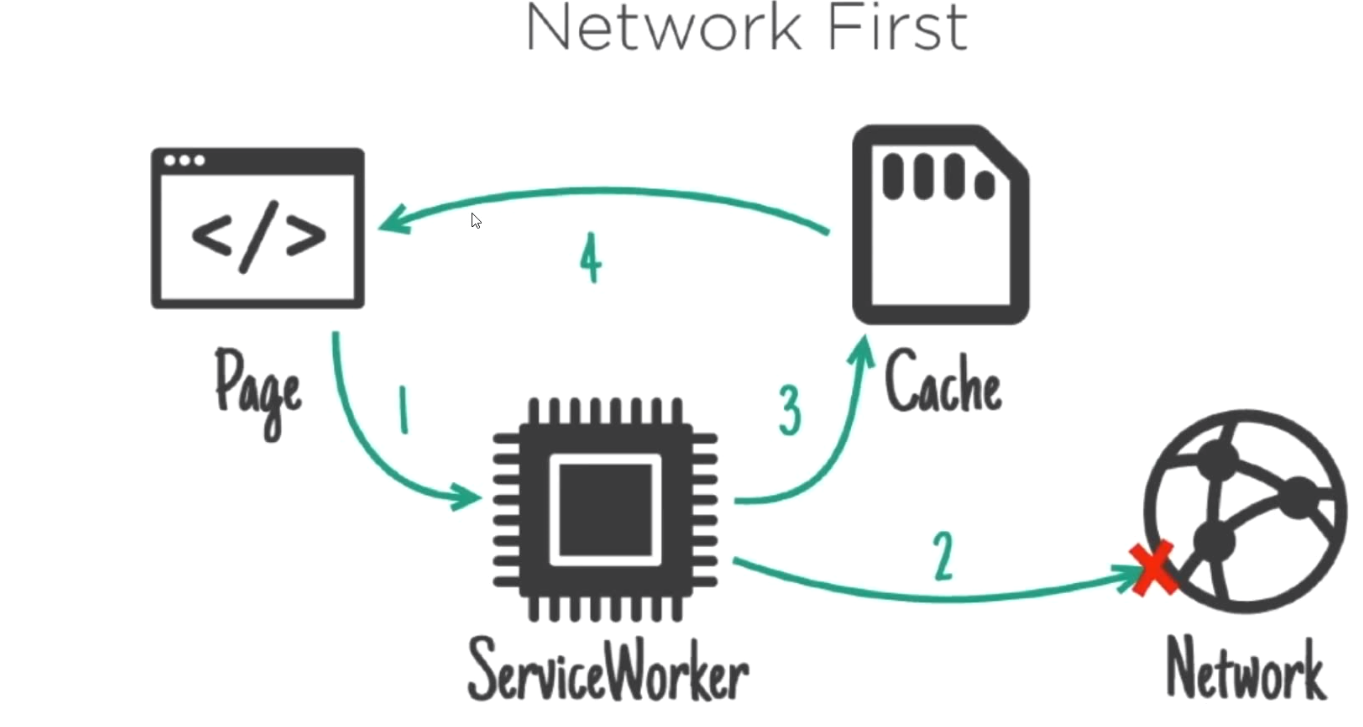
این روش ها چرخه حیات و روند عملکرد برنامه ما رو مشخص میکنه:

روش Network Only :



در تصویر بالا page برنامه ما هست، network سرور ما هست. در این روش درخواست به service worker میره و سرویس ورکر درخواست رو به نتورک میفرسته و درخواست از network به page برمیگرده. پس در اینجا کشی نداریم. اگر برنامه به صورت آفلاین باشه یا اینکه اینترنت قطع بشه برنامه ما هم قابل استفاده نخواهد بود.

روش Network First:

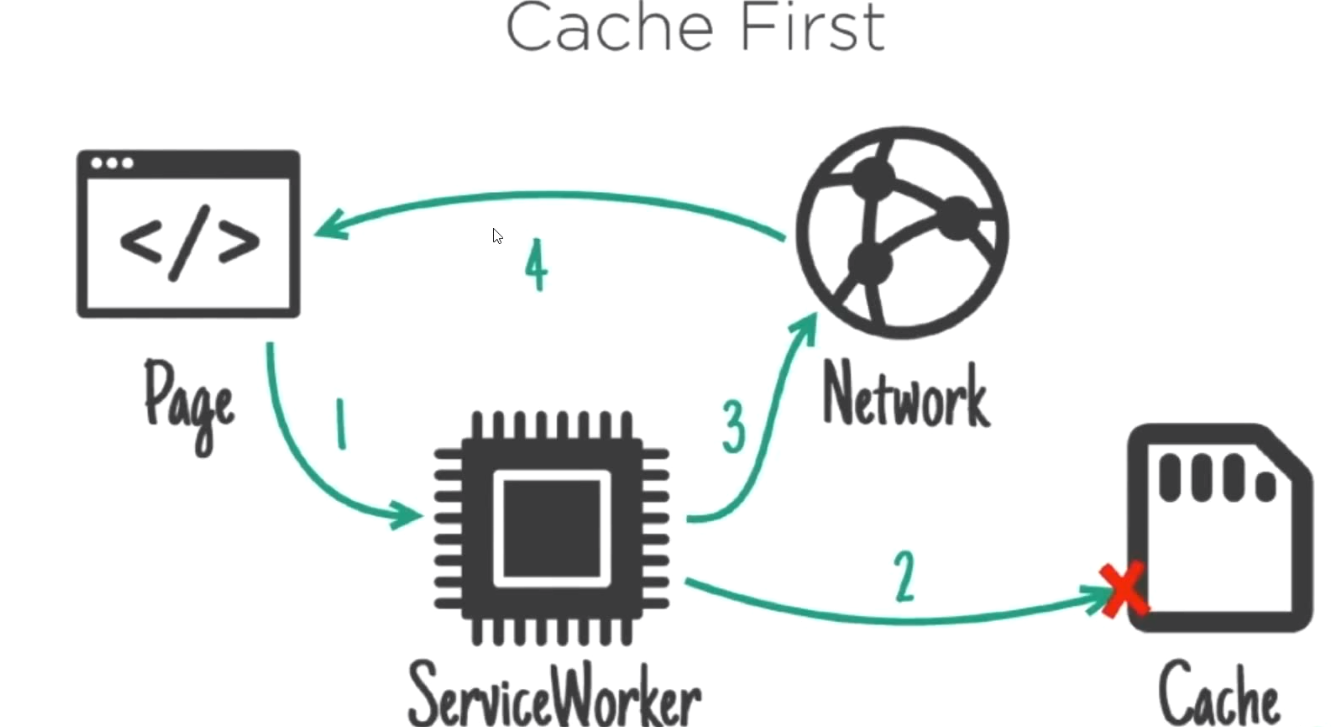


در این روش درخواست ابتدا به service worker ارسال میشه و بعد به network ارسال میشه و در صورتی که سرور به مشکل بخوره میاد سراغ cache و سعی میکنه که اطلاعات رو از chach بخونه و به page ما ارسال کنه.

گاهی اوقات این روش کارامد نیست چون در وب سایت هایی مثل بازار بورس باید همیشه آخرین وضعیت در هر لحظه رو از network داشته باشیم.

این شیوه برای وب سایت هایی مثل دیجی کالا خوب است، چون کاربر میخواهد تغییرات رو ببینه و اگر این تغییرات برای یک ساعت گذشته هم باشه اتفاقی نمی افته.

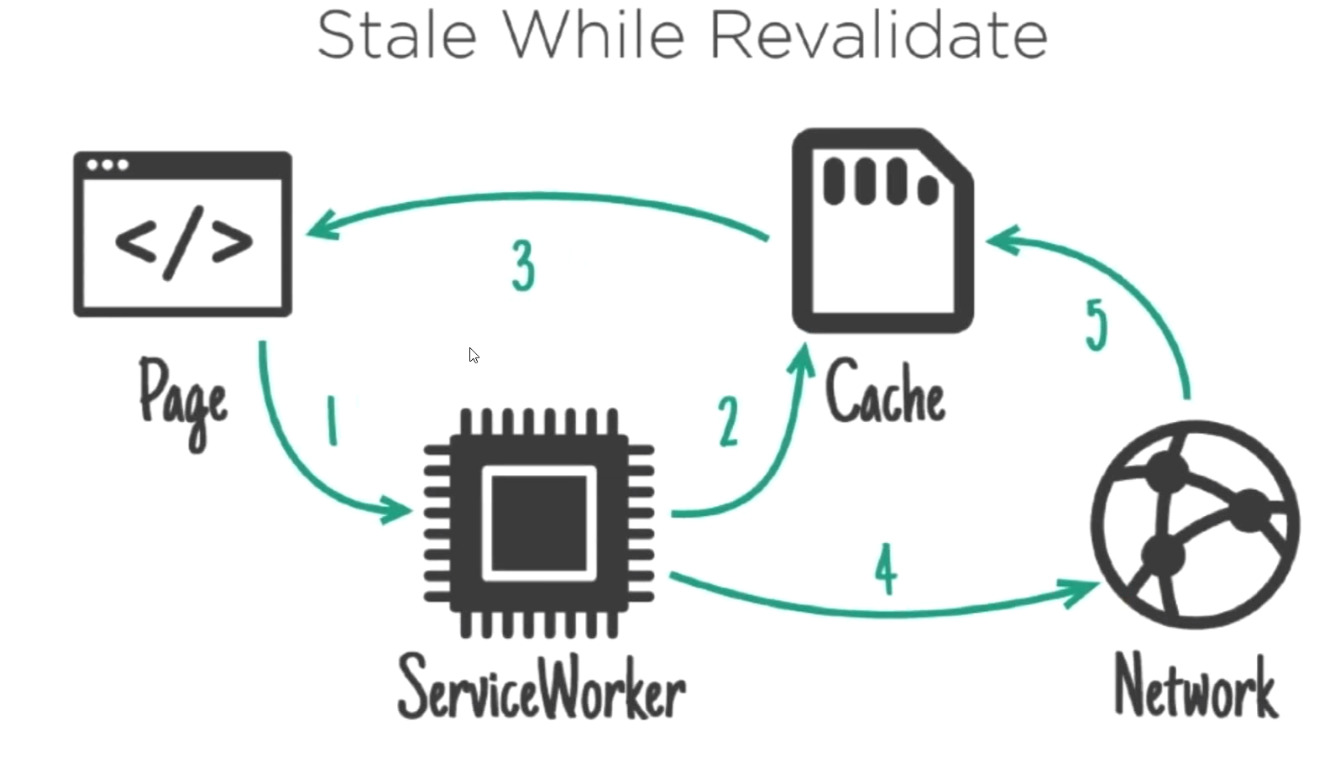
روش Cache First :



در این روش درخواست ابتدا به service worker ارسال میشه، service worker ابتدا میره سراغ cache که آیا اطلاعاتی که کاربر میخواد ببینه داخل catche هست یا نه، در صورتی که چیزی که لازم داشت در کش نبود درخواست به network ارسال میشه و نتیجه به page فرستاده میشه.

مثلا در اپلیکیشنی مثل فال حافظ فردی فالی گرفته و فایل صوتی اون رو گوش داده، اگر دوباره فال گرفت و همون فال براش اومد service worker میره به کش و میبینه فایل صوتی این فایل وجود داره و برای کاربر میارتش و دیگه به network درخواست نمیفرسته.

روش Stale While Revalidate :



درخواست ابتدا به service worker ارسال میشه و از اونجا درخواست هم به catch و هم به network ارسال میشه. ( یعنی دو درخواست همزمان به صورت asncy ارسال میشه).

و اطلاعات کش رو به page میفرسته و درخواست رو به network و اطلاعات به روز رسانی شده از network به کش فرستاده میشه و دوباره اطلاعات به روز رسانی شده رو در page خواهد دید.

در اپلیکیشن هایی مثل بانک داری به کار می آید.

برای اینکه بخواهیم یکی از این روش ها رو انتخاب کنیم باید سوالاتی رو از خودمون بپرسیم :

1. آیا میتونیم اطلاعات قدیمی رو نشون بدیم و اینکه کاربر اطلاعات قدیمی رو ببینه مشکلی پیش نمیاد؟ مثل تلگرام که وقتی اینترنت نداریم میتونیم چت های قدیمی رو ببینیم.
2. در صورتی آفلاینی امکان فالبک وجود داره؟ یعنی منتظر جواب از سمت سرور بمونیم و اطلاعات قبلی رو نشون کاربر بدیم.
3. آیا میتونیم اطلاعات قدیمی رو در حالت آفلاین نشون بدیم؟

.

.

.

**جلسه دوازدهم : ارتباط با سرور و دریافت اطلاعات**

در پروژه ای که درست کرده ایم در فایل index.js جایی که service worker خود را درست کردیم باید api خودمون رو صدا بزنیم که از fetch استفاده میکنیم.

Fetch(<“><https://toplearn.com/flights.json>”)

.then(response => response.json())

.then(data => {

let html= “”;

data.forEach(flight =>{

html += ‘<div class=’card’>${flight.city}</div>’;

});

document.duerySelector(“#screen-flights”).innerHTML = html; });

دستورات بالا اطلاعات هواپیماها رو از api که صداش زدیم میاره و در تگ با ایدی screen-flights میچینه.

برای اینکه برناممون بتونه به صورت آفلاین هم کار کنه و از کش داینامیکی که صحبت کردیم استفاده کنیم:

وارد src-sw.js میشیم :

Workbox.routing.registerRoute(

/[https:\/\/toplearn.com\/flights\.json](https://toplearn.com/flights.json)/ ,

new workbox.strategies.NetworkFirst()

)

حالا با این دستورات کش ما اتفاق می افته.