13.02.2021

**Service worker regisztrálása:**

<https://developers.google.com/web/fundamentals/codelabs/offline>

letrehoztam egy sw.js-t

service worker regisztralas, instalalalas es aktivalas (v vmi ilyesmi)

//1. instalalas  aktivalas

self.addEventListener('install', (event) => {

    self.skipWaiting();

    console.log('updated service worker installed', event);

});

self.addEventListener('activate', (event) => {

    console.log('updated service worker activated', event);

});

//This code adds an event listener to our service worker by calling addEventListener on self

//self within a service worker refers to the service worker itself).

//itself). This listener will listen to all fetch events that pass through the service worker and run the function we define next, passing it an event object as its sole argument

self.addEventListener("fetch", function(event) {

    console.log("Fetch request for:", event.request.url);

    });

//Now, instead of the browser’s native error message, you should see a personalized message

var responseContent =

"<html>" +

"<body>" +

"<style>" +

"body {text-align: center; background-color: #333; color: #eee;}" +

"</style>" +

"<h1>Elfogyott a net :( </h1>" +

"</body>" +

"</html>";

self.addEventListener("fetch", function(event) {

event.respondWith(

fetch(event.request).catch(function() {

return new Response(

responseContent,

{headers: {"Content-Type": "text/html"}}

);

})

);

});

index.html

 <script>

      (() => {

if ('serviceWorker' in navigator) {

window.addEventListener('load', () => {

navigator.serviceWorker.register('sw.js').

then((registration) => {

console.log('registered');

console.log(registration);

},(err) => {

console.log(err);

});

});

} else {

alert('No service worker support in this browser');

}

})();

      </script>

Kitöröltem az alap service worker amit a react hozott letre es annak minden meghivasat

17.02.2021

Cache-lés

Google másoltam

/\*On install - caching the application shell

We can cache the HTML, CSS, JS, and any static files that make up the application shell in

the install event of the service worker: \*/

const filesToCache = [

        '/',

        '/offline.html',

        '../src/assets/css/global.css',

        '../src/assets/images/headerimage.jpg',

        '../src/components/Recommendations.jsx',

        '../src/components/RecommendationsCarousel.jsx'

];

const staticCacheName = 'pages-cache-v1';

self.addEventListener('install', event => {

  console.log('Attempting to install service worker and cache static assets');

  event.waitUntil(

    caches.open(staticCacheName)

    .then(cache => {

      return cache.addAll(filesToCache);

    })

  );

});

self.addEventListener('activate', event => {

  console.log('Activating new service worker...');

  const cacheWhitelist = [staticCacheName];

  event.waitUntil(

    caches.keys().then(cacheNames => {

      return Promise.all(

        cacheNames.map(cacheName => {

          if (cacheWhitelist.indexOf(cacheName) === -1) {

            return caches.delete(cacheName);

          }

        })

      );

    })

  );

});

self.addEventListener('fetch', event => {

  console.log('Fetch event for ', event.request.url);

  event.respondWith(

      // Try the cache

    caches.match(event.request)

    .then(response => {

      if (response) {

        console.log('Found ', event.request.url, ' in cache');

        return response;

      }

      console.log('Network request for ', event.request.url);

      return fetch(event.request)

      .then(response => {

        return caches.open(staticCacheName)

        .then(cache => {

          cache.put(event.request.url, response.clone());

          return response;

        });

      });

    }).catch(error => {

         // If both fail, show a generic fallback:

      console.log('Error, ', error);

      return caches.match('offline.html');

    })

  );

});