Page code: console.log(navigator.serviceWorker.controller);

Workie: install and activate listener = fetch

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

console.log(event.request.url);

})

With a return(response)

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

event.respondWith(new Response("hello"));

})

With a return(resource)

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

event.respondWith(fetch(event.request.url));

})

Add Waiting Kolohe

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

console.log("new SW installed");

});

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

console.log(event.request.url);

event.respondWith(fetch(event.request.url));

});

Add Waiting Kolohe with error

self.skipWaitng();

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

throw Error("error in fetch handler");

event.respondWith(fetch(event.request.url));

});

Add waiting Kolohe with a return

self.skipWaiting();

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

return "hello workie";

});

Without refresh

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

event.waitUntil(clients.claim())

});

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

event.respondWith(fetch(event.request.url));

});

Cache

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

event.waitUntil(

caches.open("kolohe-cache")

)

})

Install

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

event.waitUntil(

caches.open("kolohe-cache")

)

})

Chimney resource to cache

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

event.waitUntil(caches.open("kolohe-cache").then(cache => {

return cache.add("/serviceworkies/buildings/stronghut/chimney.png");

}))

})

All resource to cache

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

event.waitUntil(caches.open("kolohe-cache").then(cache => {

return cache.addAll([

"/serviceworkies/buildings/stronghut/chimney.png",

"/serviceworkies/buildings/stronghut/leftside.png",

"/serviceworkies/buildings/stronghut/door.png",

"/serviceworkies/buildings/stronghut/rightside.png",

"/game/main"

])

}))

})

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

event.respondWith(

caches.open("kolohe-cache").then(cache => {

return cache.match(event.request.url).then(response => {

return response || fetch(event.request.url)

})

})

)

})

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

event.respondWith(

caches.match(event.request.url).then(response => {

return response || fetch(event.request.url);

})

)

})

Return matches via all caches

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

event.respondWith(

caches.match(event.request.url).then(response => {

return response || fetch(event.request.url);

})

)

})

Delete cache

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

event.waitUntil(

caches.delete("kolohe-cache")

)

})

Create cache with a version name on install

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

event.waitUntil(

caches.open("kolohe-cache")

) })

const version = 3;

const cacheName = `kolohe-cache-${version}`;

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

event.waitUntil(

caches.open(cacheName)

)

})

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

event.waitUntil(

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

return Promise.all(

cacheNames.map(name => {

if (name ===! cacheName){

return caches.delete(name)

}

})

)

})

)

})