Mobile dev 01

12/10/2020

Paolo Cargnin

Struttura del corso

1° lezione	2° lezione	3° lezione	4° lezione
Progressive Web apps	Progressive Web apps		

Sommario lezione

- Progressive web apps
  - Cosa sono
  - Application Manifest
  - Service Workers

Mobile APPS?

Progressive Web App

Native Apps

Mobile APPS?

Progressive Web App

Angular accede al DOM e lo passa al componente (localReference, ngModel e @ViewChild)

Native Apps

Il form è programmato direttamente da noi e sincronizzato con il DOM

Ok ma con le native APPS puoi...

- Usare le notifiche
- Avere un icona nella home del telefono
- Usare funzionalità del telefono (telecamera, bluetooth ecc)
- Lavorare offline

Guess what...

Anche le PWA possono fare queste cose

Ok ma con lo store la mia app è più scaricabile!

Ok ma con lo store la mia app è più scaricabile!

Mhhhh, proprio sicuro?

Utilizzatori per mese

Progressive Web App

8.9 MILIONI

Native Apps

3.3 MILIONI

(da comScore Mobile Metrix, dati per US)

nell’80% ovviamente teniamo conto di

- Gmail
- Facebook
- WhatsApp
- Instagram
- …

Quante app installa l’utente medio al mese?

Dalle 2 alle 5 app al mese

(da https://www.researchgate.net/)

Guardiamo un esempio:

https://app.ft.com/

Scarichiamo 01-001-pwa-base

npm i

npm start

e....

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

INIZIAMO A CAPIRCI QUALCOSA

Service Workers

Background sync

Caching

Funzionalità delle PWA

Web push

Application Manifest

Responsive Design

Ultima precisazione:

Progressive Web App

Single Page Application

Ultima precisazione:

Progressive Web App

Single Page Application

Si può fare una Spa non Pwa o una Pwa non Spa

Perchè “progressive”?

Si possono implementare le feature in maniera progressiva (e senza che una condizioni l’altra)

Cose da sapere

- Application manifest
- Service Worker
- Promises And Fetch
- Service Worker Offline access
- Strategie per gestire la cache
- Cachare dati dinamici con IndexedDB
- Background Sync
- Web Push Notification
- Media API (camera) e Geolocation
- Workbox

Vediamo dove arriviamo!

In caso... continuiamo domani (?)

Iniziamo la nostra APP!
01-002-pwa-personale

npm i

npm start

WEB APP MANIFEST

Rende la tua app installabile

nella home screen

Creiamo il file manifest.json

Nella cartella *public*
Inseriamo poi come
`<link rel="manifest" href="/manifes.json">`

Contenuto

```
{
  "name": "PWA gram", //  Nome lungo dell'app (splashscreen)
  "short_name": "PWAGram", //  Nome breve dell'app (sotto l'icon)
  "start_url": "/index.html", // Quale file caricare all'inizio
  "scope": ".", // Quali sono le pagine del sito che fanno parte della PWA
  "display": "standalone",  // standalone - fullscreen - minimal-ui - browser
  "background_color": "#fff", // Background di splashscreen
  "theme_color": "#3f51b5", // Colore principale dell'app
  "description": "Un clone in PWA di Instagram", // non è usato, ma magari in futuro....
  "dir": "ltr", // left to right o right to left?
  "lang": "en-US", // Lingua di default
  "orientation": "portrait-primary", // any - landscape - portrait - portrait-primary
  "icons": [
    {
      "src": "/src/images/icons/app-icon-48x48.png",
      "type": "image/png",
      "sizes": "48x48"
    }
    ...
  ]
}
```

Ultima proprietà

```
{
    "related_applications": [
      {
        "platform": "play",
        "url": "....",
        "id": "com.example.app1"
      }
      ...
    ]
}
```

Can I use it?

https://caniuse.com/web-app-manifest

Cose da tenere in mente per usare un emulatore

10.0.2.2 è il localhost della nostra macchina

Oppure...

https://developers.google.com/web/tools/chrome-devtools/remote-debugging/

Aiutiamo Safari ad avere un aspetto simile

```
<meta name="apple-mobile-web-app-capable" content="yes">
<meta name="apple-mobile-web-app-status-bar-style" content="black">
<meta name="apple-mobile-web-app-title" content="PWAGram">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-57x57.png" sizes="57x57">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-60x60.png" sizes="60x60">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-72x72.png" sizes="72x72">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-76x76.png" sizes="76x76">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-114x114.png" sizes="114x114">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-120x120.png" sizes="120x120">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-144x144.png" sizes="144x144">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-152x152.png" sizes="152x152">
<link rel="apple-touch-icon" href="/src/images/icons/apple-icon-180x180.png" sizes="180x180">
```

Aiutiamo Internet explorer

```
<meta name="msapplication-TileImage" content="/src/images/icons/app-icon-144x144.png">
<meta name="msapplication-TileColor" content="#fff">
```

Meta importante

```
<meta name="theme-color" content="#3f51b5">
```

Service Worker

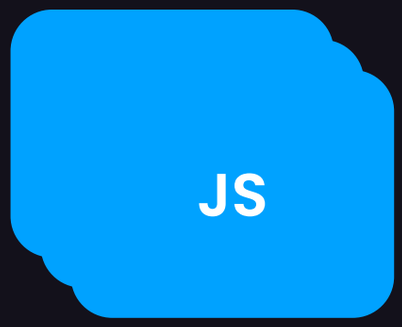
Gestione della cache

Background Sync

Push Notification

Fanno del lavoro “di nascosto”

Cosa Sono?



Viene eseguito in un singolo thread, attacato a singole pagine html

Processi in background!

JS

Viene eseguito in un singolo thread, attacato a singole pagine html

Single thread

HTML file

HTML file

HTML file

SERVICE WORKER

Viene eseguito in un altro thread, **distaccato** dalle pagine HTML!
Gestisce tutte le pagine di uno scope (per esempio, di un dominio)
Continua ad essere eseguito anche quando tutte le pagine sono state chiuse

Single thread

Cosa Sono?

processi in background

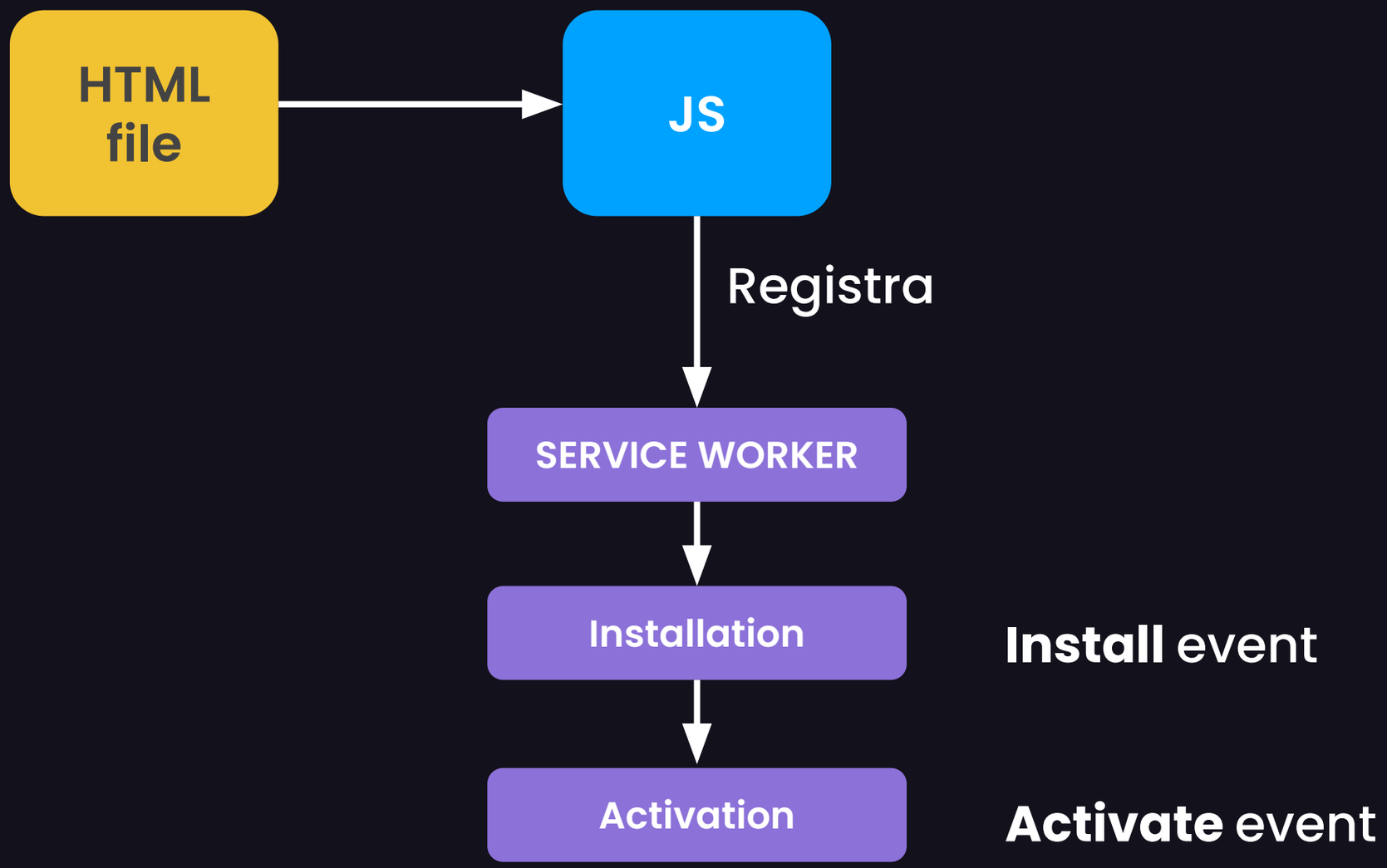
che reagiscono ad eventi

(dall'HTML, da un altro server ecc)

Eventi che può ascoltare

Fetch	Browser o JS di una pagina fa un fetch (HTTP request) Immagini, css ecc - NO XML XHR (ajax old http request)
Push Notifications	Service Worker riceve Web Push Notification (from Server)
Notification interaction	L’utente interagisce con una notifica
Background Sync	Il service Worker ricee un background Sync Event (esempio: Sei di nuovo connesso)
SW lifecycle	Fasi del Service worker

Service Worker lifecycle



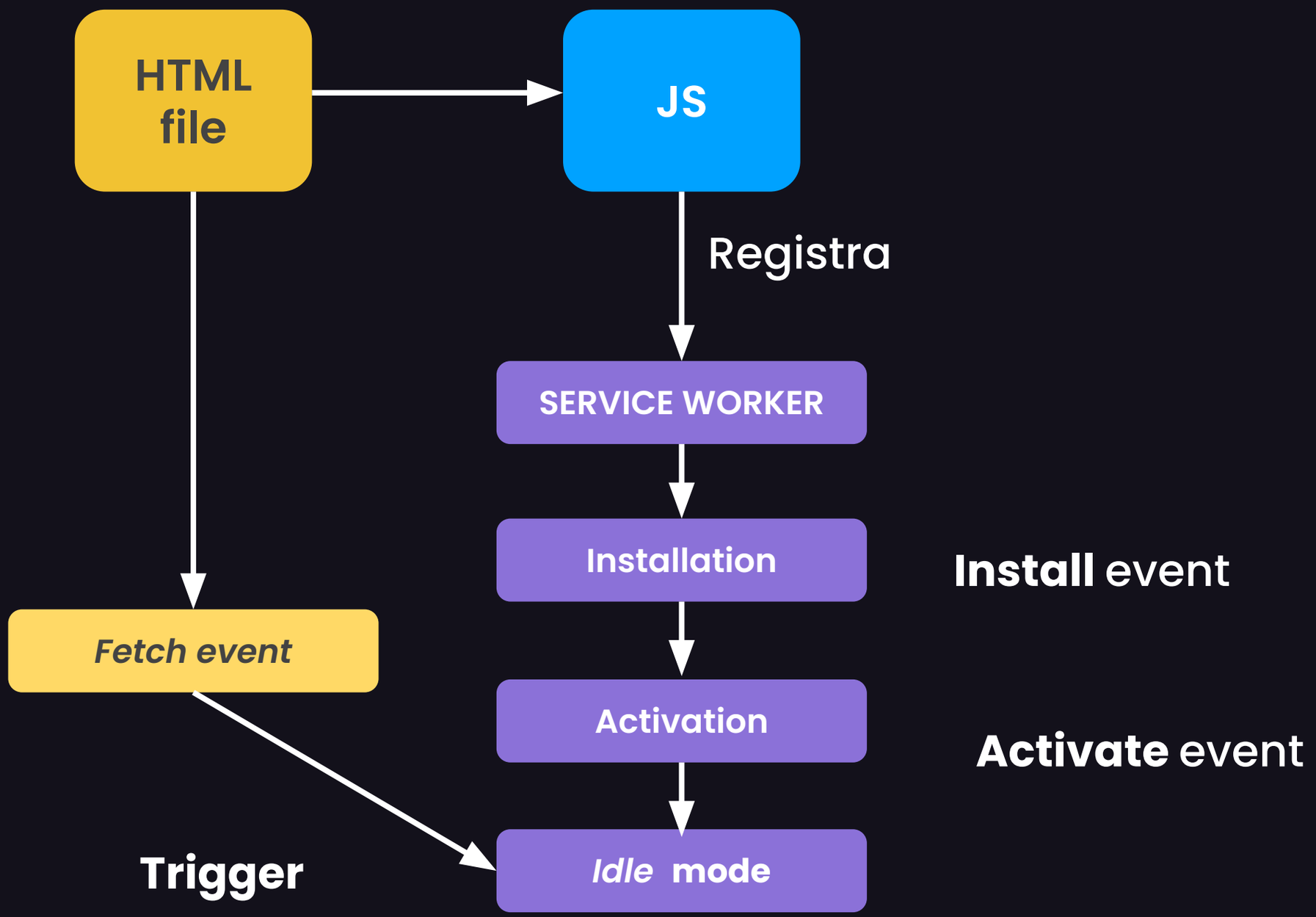
Da adesso, il service worker controlla tutte le pagine dello scope

Ogni volta che carichiamo una nuova pagina

Viene generato un nuovo Service Worker!?

NO!

Perchè **L’install** event viene eseguito solo se il file js che ha creato il nostro serviceWorker cambia di un Byte o più.



Da adesso, il service worker controlla tutte le pagine dello scope

Service Worker è supportato da?

https://caniuse.com/serviceworkers

Piccola nota,

Quando si lavora coi service worker in locale (i file cambiano spesso)

Devi assicurarti che la cache del browser non venga salvata, nel nostro esempio:

```
...
"scripts": {
    "start": "http-server -c-1"
},
...
```

Il serviceWorker agisce solo nelle pagine della cartella in cui si trova

E nelle sottocartelle

Iniziamo!

Creiamo il file

sw.js in public

Registriamolo in un file JS
meglio registrarlo in un file globale.

app.js

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker
    .register('/sw.js')
    .then(function() {
      console.log('Service worker registered!');
    });
}
```

Puoi usare scope per dirgli che file può guardare

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker
    .register('/sw.js',{scope: '/'})
    .then(function() {
      console.log('Service worker registered!');
    });
}
```

Vediamo i nostri eventi

```
self.addEventListener('install', function(event) {
  console.log('[Service Worker] Installing Service Worker ...', event);
});
```

Activate

```
self.addEventListener('activate', function(event) {
  console.log('[Service Worker] Activating Service Worker ...', event);
  return self.clients.claim(); //
});
```

Solo activate?

Vediamo perché

SUPER IMPORTANTE!

Ultimo evento...

```
self.addEventListener('fetch', function(event) {
  console.log('[Service Worker] Fetching something ....', event);
  //Prova a mettere null al posto di // fetch(event.request)
  event.respondWith(fetch(event.request));
});
```

Ora, capiamo come possiamo installare la APP in locale!

Facciamolo in un sito qualsiasi

Come far si che chrome chieda di scaricare la app?

https://googlechrome.github.io/samples/app-install-banner/

e

https://web.dev/customize-install/

Customizziamo il popup

in app.js:

```
var deferredPrompt;

window.addEventListener('beforeinstallprompt', function(event) {
  console.log('beforeinstallprompt!);
  event.preventDefault();
  deferredPrompt = event;
  return false;
});
```

Customizziamo il popup

in feed.js:

```
function openCreatePostModal() {
  createPostArea.style.display = 'block';
  if (deferredPrompt) {
    deferredPrompt.prompt();

    deferredPrompt.userChoice.then(function(choiceResult) {
      console.log(choiceResult.outcome);

      if (choiceResult.outcome === 'dismissed') {
        console.log('User cancelled installation');
      } else {
        console.log('User added to home screen');
      }
    });

    deferredPrompt = null;
  }
}
```

F.A.Q. TIME!

Il serviceWorker si installa ogni volte che ricarico la pagina?

No. Solo quando il file che lo ha inizializzato cambia di almeno un byte.

F.A.Q. TIME!

Posso cancellare via codice un serviceWorker?

Certo, così:

```
navigator.serviceWorker.getRegistrations().then(function(registrations) {
  for(let registration of registrations) {
   registration.unregister()
 } })
```

F.A.Q. TIME!

Si possono avere diversi ServiceWorker per una pagina?

Si ma è molto raro, e comunque non possono avere lo stesso scope.

```
navigator.serviceWorker.getRegistrations().then(function(registrations) {
  for(let registration of registrations) {
   registration.unregister()
 } })
```

F.A.Q. TIME!

Il mio serviceWorker può mandare messagi alle mie pagine web?

Certo, usando i messaggi:

https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API/Using_web_workers#Sending_messages_to_and_from_a_dedicated_worker

F.A.Q. TIME!

Il mio serviceWorker può mandare messagi alle mie pagine web?

Certo, usando i messaggi:

https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API/Using_web_workers#Sending_messages_to_and_from_a_dedicated_worker

Capitoletto veloce:

Promises e Fetch

Promises e Fetch

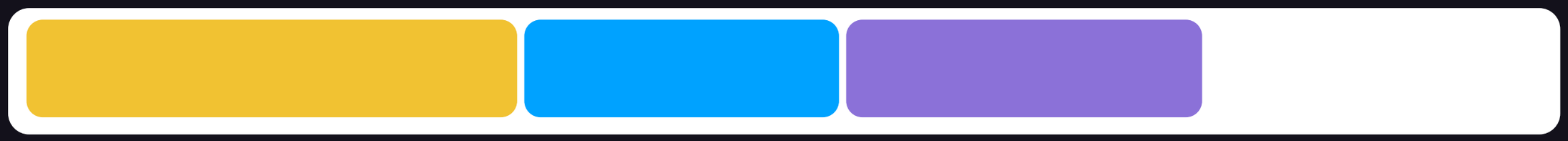
JavaScript è single Thread!

Fetch Data from server

setTimeout

Altri task..

Ma non funziona così:



Promises e Fetch

JavaScript è single Thread!

Fetch Data from server

setTimeout

Altri task..

Ma così:

Init Async Task

Tempo per altre operezioni

Riscriviamo setTimeout con le Promise

Vediamo .then() e .catch()

Fetch

Fetch è un metodo messo a disposizione dal Browser

Sostituisce il vecchio

“new XMLHttpRequest()”

Usiamola

```
fetch('https://pokeapi.co/api/v2/pokemon/volpix')
  .then( () => {
    console.log(response)
  })
```

Readable

```
fetch('https://pokeapi.co/api/v2/pokemon/volpix')
.then( (result) => {
  console.log(result)
  return result.json
})
.then((data) => {
  console.log(data)
})
```

Catch

```
fetch('https://pokeapi.co/api/v2/pokemon/volpix')
  .then( (result) => {
    console.log(result)
    return result.json
  })
.then((data) => {
  console.log(data)
})
.catch(e => {
  console.error(e)
})
```

POST request?

```
fetch('https://pokeapi.co/api/v2/pokemon/volpix',{
  method: 'POST',
  header: {
    'Content-Type': 'application/json',
    'Accept':  'application/json'
  },
  body: JSON.stringify({foo: "bar"})
})
  .then( (result) => {
    console.log(result)
    return result.json
  })
  .then((data) => {
    console.log(data)
  })
  .catch(e => {
    console.error(e)
  })
```

Fetch vs Ajax

```
fetch('https://pokeapi.co/api/v2/pokemon/volpix')
  .then( (result) => {
    console.log(result)
    return result.json
})
.then((data) => {
    console.log(data)
})
.catch(e => {
    console.error(e)
})
```

```
var xhr = new XMLHttpRequest();
xhr.open('GET', 'https://httpbin.org/ip');
xhr.responseType = 'json';

xhr.onload = function() {
  console.log(xhr.response);
};

xhr.onerror = function() {
  console.log('Error!');
};

xhr.send();
```

I ServiceWorker funzionano solo con Fetch

Se il browser no li supporta

Dobbiamo usare i polyfills:

https://github.com/github/fetch

https://github.com/taylorhakes/promise-polyfill

Service Worker - Caching!

rendiamo l’applicazione offline-capable

Perché farlo?

Non sempre abbiamo la connessione

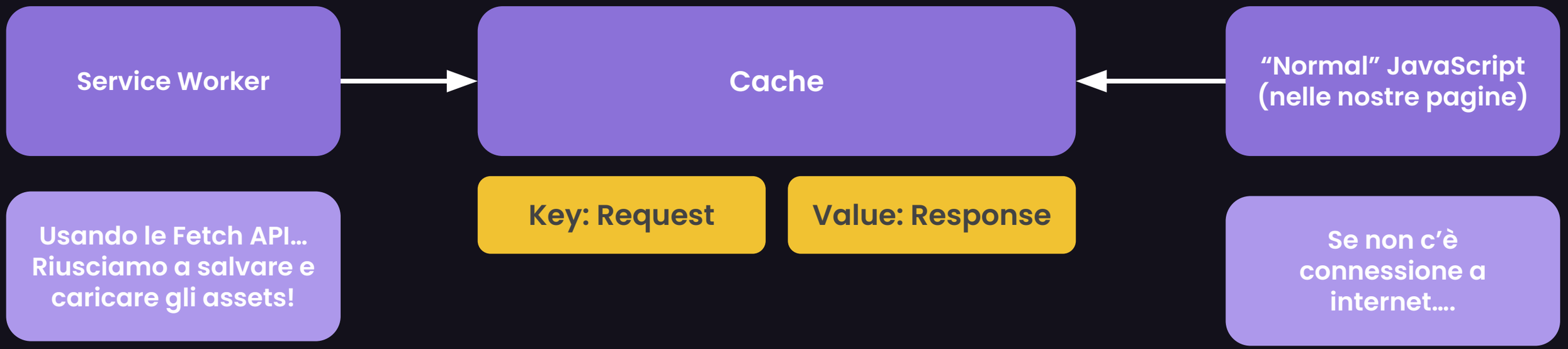
Anche quando dobbiamo eseguire semplici operazioni!

Le Cache API

La cache del server, ci è inutile in questo caso.

Di default, ci sarebbe la cache del browser, ma non è possibile gestirla.

Le Cache API


Service Worker
Cache
“Normal” JavaScript (nelle nostre pagine)
Key: Request
Value: Response
Usando le Fetch API… Riusciamo a salvare e caricare gli assets!
Se non c’è connessione a internet….

Can I Use?

https://caniuse.com/?search=cache

Prepariamo il nostro progetto

Caricate 01-004-caching

npm i

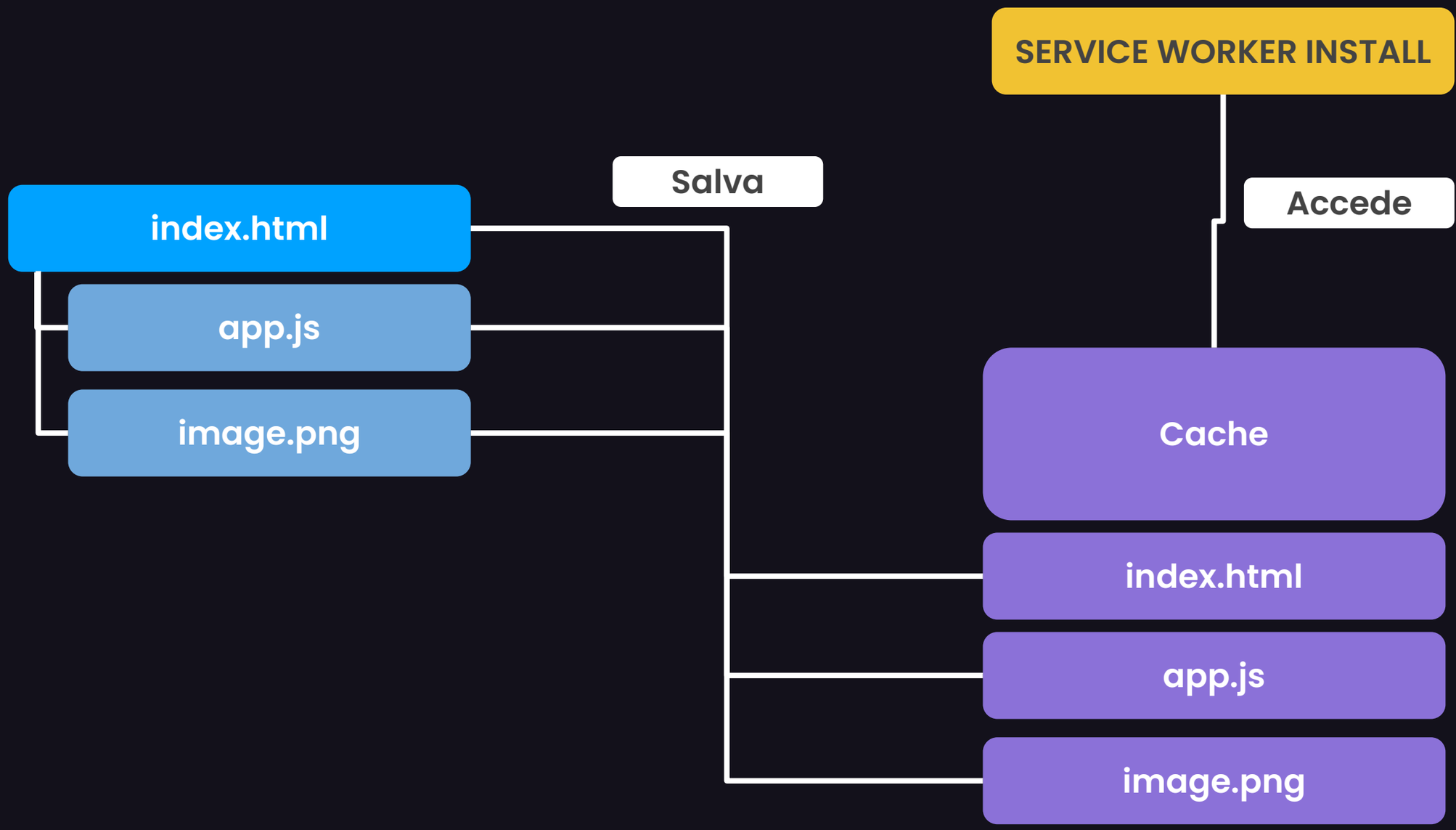
npm start

Non funziona?

npm install --save-dev http-server@0.12.3

Static Caching

durante installazione



Guardiamo i file statici della nostra applicazione

stylesheet, javascripts...Fonts! :/

in sw.js

```
self.addEventListener('install', function(event) {
  console.log('[Service Worker] Installing Service Worker ...', event);
  caches.open();
});
```

meglio così:

```
self.addEventListener('install', function(event) {
  console.log('[Service Worker] Installing Service Worker ...', event);
  event.waitUntil(caches.open());
});
```

Aggiungiamo un file!

```
self.addEventListener('install', function(event) {
  console.log('[Service Worker] Installing Service Worker ...', event);
  event.waitUntil(
    caches.open('static')
    .then(function(cache) {
      console.log('pre caching')
      cache.add('/src/app.js')
    })
  );
});
```

Metodi di cache

https://developer.mozilla.org/en-US/docs/Web/API/Cache

Guardiamo il cacheStorage su chrome

non siamo ancora offline...

Carichiamo le risorse Cachizzate!

Ovviamente, ci metteremo in mezzo all’evento...

fetch!

in sw.js

```
self.addEventListener('fetch', function(event) {
  event.respondWith(
    caches.match(event.request)
      .then(function(response) {
        if (response) {
          return response
        } else {
          return fetch(event.request)
        }
      })
  );
});
```

Aggiungiamo il file html

/index.html

Non funziona!

mh. in realtà si, andiamo su /index.html sul browser

Aggiungiamo quindi

/

Aspetta, dobbiamo aggiungere tutti gli url?

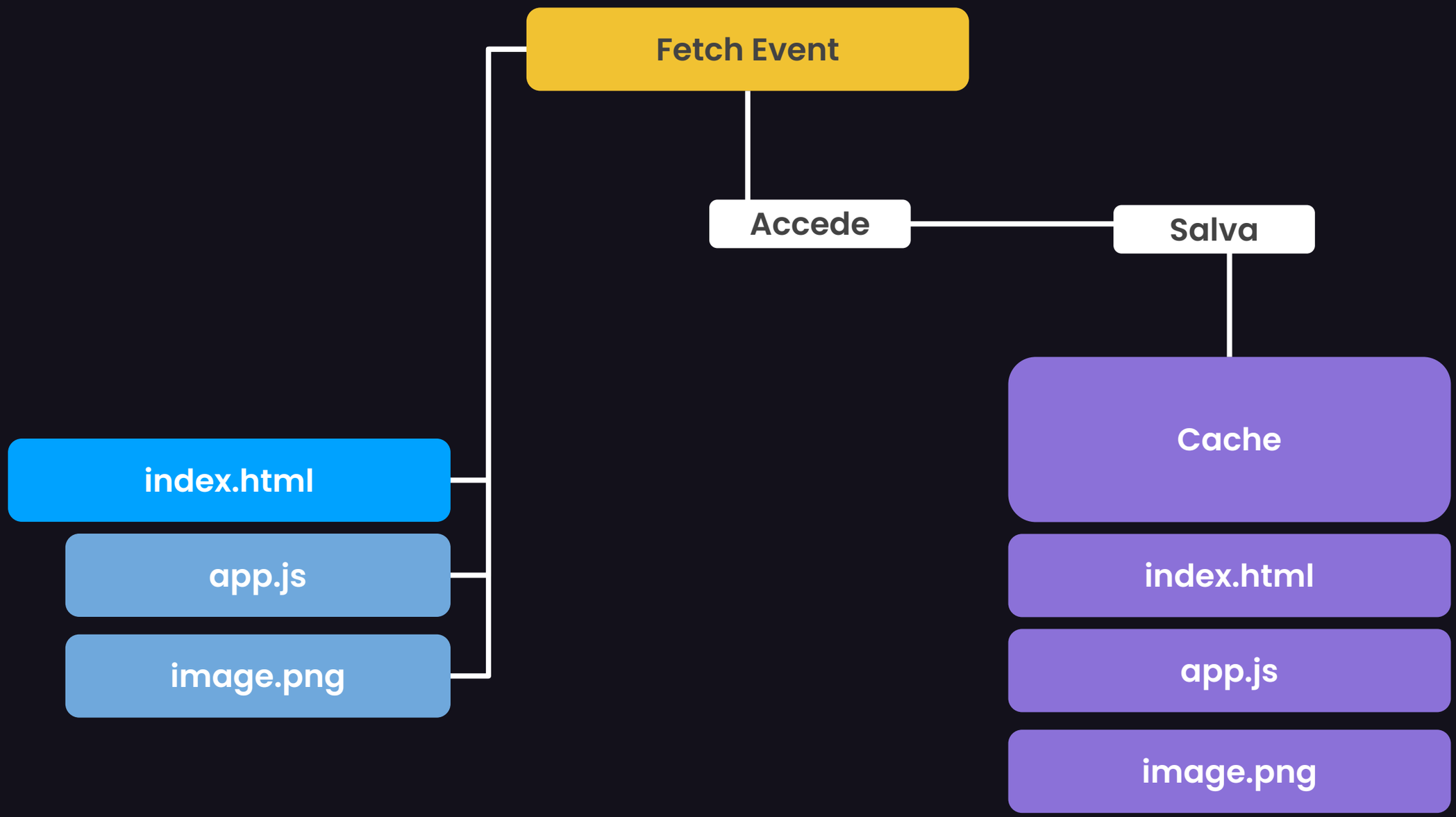
addAll does the trick!

```
cache.addAll([
  '/',
  '/index.html',
  '/src/js/app.js',
  '/src/js/feed.js',
  '/src/js/promise.js',
  '/src/js/fetch.js',
  '/src/js/material.min.js',
  '/src/css/app.css',
  '/src/css/feed.css',
  '/src/images/main-image.jpg',
  'https://fonts.googleapis.com/css?family=Roboto:400,700',
  'https://fonts.googleapis.com/icon?family=Material+Icons',
  'https://cdnjs.cloudflare.com/ajax/libs/material-design-lite/1.3.0/material.indigo-pink.min.css'
]);
```

Ok. Non è possibile gestire un applicativo così però

1. Le icone stanno funzionando offline?
2. Come faccio a mantenere il tutto?

Dynamic Cache!



Modifichiamo il SW!

```
self.addEventListener('fetch', function(event) {
  event.respondWith(
    caches.match(event.request)
      .then(function(response) {
        if (response) {
          return response;
        } else {
          return fetch(event.request)
            .then(function(res) {
              return caches.open('dynamic')
                .then(function(cache) {
                  cache.put(event.request.url, res.clone());
                  return res;
                })
            })
            .catch(function(err) {

            });
        }
      })
  );
});
```

Gestiamo gli errori!

```
self.addEventListener('fetch', function(event) {
  event.respondWith(
    caches.match(event.request)
      .then(function(response) {
        if (response) {
          return response;
        } else {
          return fetch(event.request)
            .then(function(res) {
              return caches.open('dynamic')
                .then(function(cache) {
                  cache.put(event.request.url, res.clone());
                  return res;
                })
            })
            .catch(function(err) {

            });
        }
      })
  );
});
```

Rimane ancora un errore

è il nostro sw.js

Quindi ha senso.

Piccola nota

Così chachiamo anche le chiamate HTTP relative ai dati. Non è il massimo, poi vediamo come toglierle

Come versioniamo la nostra cache?

Usiamo delle variabili per il nome della cache!

`CACHE_STATIC_NAME` e `CACHE_DYNAMIC_NAME`

Puliamo la vecchia cache

Quando lo facciamo?

activate

In sw.js

```
self.addEventListener('activate', function(event) {
  console.log('[Service Worker] Activating Service Worker ....', event);
  event.waitUntil(
    caches.keys()
      .then(function(keyList) {
        return Promise.all(keyList.map(function(key) {
          if (key !== CACHE_STATIC_NAME && key !== CACHE_DYNAMIC_NAME) {
            console.log('[Service Worker] Removing old cache.', key);
            return caches.delete(key);
          }
        }));
      })
  );
  return self.clients.claim();
});
```

Riassunto!

- Cache Api
- cache.add
- chache.addAll
- cache.delete

Esercizio!

01-005-caching-esercizio