LDD

> Lighthouse driven development (From SPA to PWA)

SPA

JS basics

- DOM manipulation
- Array methods (map, filter, reduce)
- Fetch
- Promises

What is Lighthouse

- Audits tab in Google Chrome
- Chrome Extension
- CLI

What is a PWA

- Responsive
- Connectivity-independent
- Interactive with a feel like a native app's
- Always up-to-date
- Safe
- Discoverable
- Re-engageable
- Installable
- Linkable

Service Workers

- Runs in its own global script context
- No DOM access
- HTTPS only
- It's a separate file
- It need to be registered

Registration

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker
    .register('sw.js')
    .then(() => {
      console.log('Oh yeah mate ✌');
    })
    .catch(err => {
      console.log('☠ Oh no!', err);
    });
}
```

Intercept requests

- The fetch event

```
self.addEventListener('fetch', e => {
  e.respondWith(
    caches.match(e.request).then(response => {
      return response || fetch(e.request);
    })
  );
});
```

Lifecycle

- see slides

Cache API

- differencies between Browser Cache

```
self.addEventListener('install', function(e) {
  e.waitUntil(
    caches.open('mycache-v1').then(function(cache) {
      return cache.addAll(['/', '/src/main.css',
'/src/main.min.js']);
    })
  );
});
```

Clean old caches

```
self.addEventListener('activate', e => {
  e.waitUntil(
    caches.keys().then(keyList =>
      Promise.all(
        keyList.map(key => {
          if (key !== cacheName) {
            console.log('Removing cache:', key);
            return caches.delete(key);
          }
        })
      )
    )
  );
});
```

Application Shell

Add to home screen

- You need a manifest.json file
- Your manifest file needs a start URL
- You need a 144 x 144 PNG icon

The web app manifest

App manifest generator

```
{
  "name": "Progressive web app",
  "short_name": "PWA",
  "start_url": "/index.html",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#000000",
  "icons": [
    {
      "src": "homescreen.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "homescreen-144.png",
      "sizes": "144x144",
      "type": "image/png"
    }
  ]
}
```

```
<link rel="manifest" href="/manifest.json">
```

LDD

- create a SW
- register the SW
- cache the home page
- create a manifest.json
- add theme colors, splash screen

Push Notifications

- Push Companion

Feature detect

```
if ('serviceWorker' in navigator && 'PushManager' in
window) {
}
```

Firebase Cloud Messaging

- Create a project on Firebase

Import Firebase SDK

```
yarn add @firebase/app @firebase/messaging
```

```
// notifications.js

import firebase from '@firebase/app';
import '@firebase/messaging';
```

- Settings > Cloud Messaging

Add the sender ID to the manifest.json

```
{
  "gcm_sender_id": "SENDER_ID"
}
```

Generate Key Pair

- Settings > Cloud Messaging > Web configuration

```
firebase.initializeApp({
  messagingSenderId: 'SENDER_ID'
});

const messaging = firebase.messaging();

messaging.usePublicVapidKey(PUBLIC_KEY);

messaging
  .requestPermission()
  .then(() => {
    console.log('Notification permission granted');
    // TODO Retrieve an Instance ID token for use
with FCM.
  })
  .catch(err => {
    console.log('Unable to get permission', err);
  });
```

- Create the firebase-messaging-sw.js file.
- replace the // TODO part

```
return messaging.getToken();
```

- add .then() and console log the token
- create firebase-messaging-sw.js and update webpack copy

```
// firebase-messaging-sw.js

importScripts('https://www.gstatic.com/firebasejs/4.1
.3/firebase-app.js');
importScripts('https://www.gstatic.com/firebasejs/4.1
.3/firebase-messaging.js');

firebase.initializeApp({
  messagingSenderId: SENDER_ID
});

const messaging = firebase.messaging();

messaging.setBackgroundMessageHandler(payload => {
  console.log(
    '[firebase-messaging-sw.js] Received background
message ',
    payload
  );
  return self.registration.showNotification(
    notificationTitle,
    notificationOptions
  );
});
```

Pass the generated token as key

```
curl -X POST -H "Authorization:
key=AAAAyNTjBhE:APA91bHTE7-AGbudM4Tqhx9XlHejUop-
LOUfqF7B2XHHJFlTEvKMT6xbNP2d2v73CXgpTzref8cMD7fy606KA
0JqaK6_aaGOM91hh0SrIE-RJ0Dx3uvgjBmH9TNhy_UZBR-G-
4WHL5KK" -H "Content-Type: application/json" -d '{
  "notification": {
    "title": "Hello!",
    "body": "This is my first Message",
  },
  "to":
"eDusaLgjcg8:APA91bHOulqekmoZCBq0MU2wbZR5X9FVCGJrb_To
2UyxbEHKzPYWSJOlumnsaZDj0rd4c6q5TCmSl2K7Zs4aYBEfL-
2hpsISY7QmvfdERmUfsB1s50or94GHizhDlUhemR-wzXhpzIAd"
}' "https://fcm.googleapis.com/fcm/send"
```

Notification opt-in

```
const subscribe = reg => {
  reg.pushManager.subscribe({ userVisibleOnly: true
});
};
```

Debugging Service Workers

Tools

- Workbox
- Firebase

BackgroundSync

Are PWA ready?