

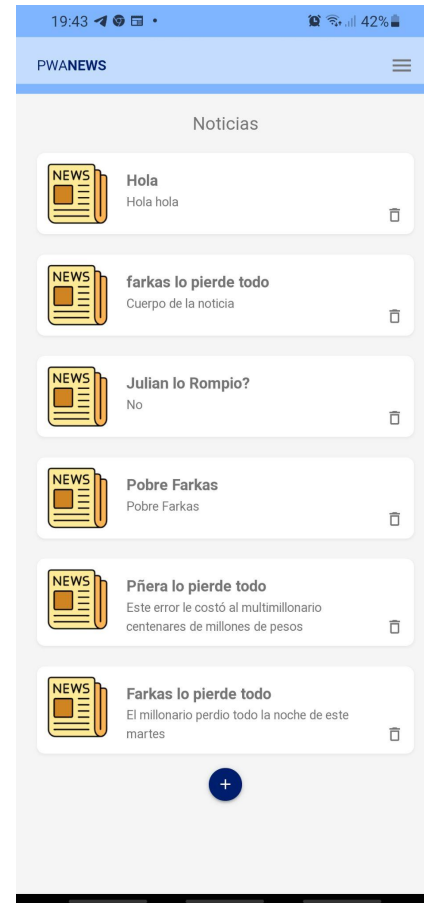


Progressive Web Apps

Grupo 1:
Joaquín Cáceres
Julián García
Mathias Valdebenito

Requerimientos para ser PWA

1. Web App Manifest — `manifest.json`
2. service worker con al menos un evento *fetch* — `serviceworker.js`
3. Iconos y tema / color de fondo (splash screen)
4. HTTPS



01

Demo: firebase app

GO GO GO!

<https://tinyurl.com/grupo-01-1>

02

Demo: app from scratch

GO GO GO!

<https://tinyurl.com/grupo-01-2>

03

Aplicación en firebase

Aplicación en firebase: manifest.json

```
{
  "name": "PWA News",
  "short_name": "News",
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#C4DDFF",
  "theme_color": "#7FB5FF",
  "orientation": "portrait-primary",
  "icons": [
    {
      "src": "/img/icons/icon-16x16.png",
      "type": "image/png",
      "sizes": "16x16"
    },
    {
      "src": "/img/icons/icon-24x24.png",
      "type": "image/png",
      "sizes": "24x24"
    },
    {
      "src": "/img/icons/icon-32x32.png",
      "type": "image/png",
      "sizes": "32x32"
    },
    ...
  ]
}
```



```
if ('serviceWorker' in navigator) {  
  navigator.serviceWorker.register('/sw.js')  
    .then((reg) => console.log('service worker registered!', reg))  
    .catch((err) => console.log('service worker not registered', err));  
}
```


Aplicación en firebase: service worker

```
const staticCacheName = 'site-static-v8';
const dynamicCacheName = 'site-dynamic-v8';
const assets = [
  '/',
  '/index.html',
  '/js/app.js',
  '/js/ui.js',
  '/js/materialize.min.js',
  '/css/styles.css',
  '/css/materialize.min.css',
  '/img/news.png',
  'https://fonts.googleapis.com/icon?family=Material+Icons',
  'https://fonts.gstatic.com/s/materialicons/v128/flUhRq6tzZclQEJ-Vdg-
  IuiaDsNaqBfQlWbafR!html'
]

// cache size limit function
const limitCacheSize = (name, size) => {
  caches.open(name).then(cache => {
    cache.keys().then(keys => {
      if (keys.length > size) {
        cache.delete(keys[0]).then(limitCacheSize(name, size))
      }
    })
  })
}
```

Aplicación en firebase: service worker

```
self.addEventListener('install', evt => {
  // console.log('service worker has been installed');
  evt.waitUntil(
    caches.open(staticCacheName).then(cache => {
      // console.log('caching shell assets')
      cache.addAll(assets)
    })
  )
});


// activate service
self.addEventListener('activate', evt => {
  // console.log('service worker has been activated');
  evt.waitUntil(
    caches.keys().then(keys => {
      // console.log(keys);
      return Promise.all(keys
        .filter(key => key !== staticCacheName && key !==
dynamicCacheName)map(key => caches.delete(key)))
    })
  )
});
```

Aplicación en firebase: service worker

```
self.addEventListener('fetch', evt => {
  if (evt.request.url.indexOf('firebase.googleapis.com') === -1) {
    evt.respondWith(
      caches.match(evt.request).then(cacheRes => {
        return cacheRes || fetch(evt.request).then(fetchRes => {
          return caches.open(dynamicCacheName).then(cache => {
            cache.put(evt.request.url, fetchRes.clone()).then(() => {
              limitCacheSize(dynamicCacheName, 15)
            });
            return fetchRes;
          })
        })
      }).catch(() => {
        if (evt.request.url.indexOf('.html') > -1) {
          return caches.match('/pages/fallback.html')
        }
      })
    )
  }
})
```

Aplicación en firebase: Inicialización de firebase

```
// Initialize Firebase
firebase.initializeApp(firebaseConfig);
const db = firebase.firestore();
const messaging = firebase.messaging();
messaging
  .requestPermission()
  .then(() => {
    return messaging.getToken();
  })
  .then(token => {
    const tokenData = {
      token,
    };
    db.collection('tokens').doc(token).set(tokenData, {merge: true})
      .catch(err => console.log(err));
  })
  .catch(err => {
    console.log("No permission to send push", err);
  });
messaging.onMessage(payload => {
  console.log("Message received. ", payload);
  const { title, ... options } = payload.notification;
});
```



```
db.enablePersistence()  
  .catch(err => {  
    if (error.code === 'failed-precondition') {  
      // probably multiple tabs open at once  
      console.log('persistence failed')  
    } else if (err.code === 'unimplemented') {  
      // lack of browser support  
      console.log('persistence is not available')  
    }  
  })  
})
```

```
// real-time listener
db.collection('news').onSnapshot(snapshot => {
  snapshot.docChanges().forEach(change => {
    // console.log(change, change.doc.data(), change.doc.id);
    if (change.type === 'added') {
      // add the document data to the web page
      renderNew(change.doc.data(), change.doc.id)
    }
    if (change.type === 'removed') {
      // remove the document data from the web page
      removeNew(change.doc.id);
    }
  })
})
```

```
const bodyToSend = {
  notification: {
    title: form.title.value,
    body: form.body.value,
    icon: '/img/icons/icon-128x128.png'
  },
  android: {
    priority: "normal"
  },
  registration_ids: tokens
}

fetch('https://fcm.googleapis.com/fcm/send', {
  method: 'POST',
  headers: {
    Authorization,
    'Content-Type': 'application/json',
  },
  body: JSON.stringify(bodyToSend)
}).catch(() => console.log('error sending notification'))
```



```
importScripts("https://www.gstatic.com/firebasejs/6.0.1/firebase-app.js", "https://www.gstatic.com/firebasejs/6.0.1/firebase-messaging.js")

firebase.initializeApp({messagingSenderId: "559509250006"});
const messaging = firebase.messaging();

messaging.setBackgroundMessageHandler(payload => {
  const notification = JSON.parse(payload.data.notification);
  const notificationTitle = notification.title;
  const notificationOptions = {
    body: notification.body
  };
  //Show the notification :)
  return self.registration.showNotification(
    notificationTitle,
    notificationOptions
  );
});
```


04

Aplicación desde cero

Aplicación desde cero: push notifications (server)



```
$ npm i web-push
```

```
$ npx web-push generate-vapid-keys
```

Public Key:

BMvnq-StnCV3TdJhioixgD7G0MtmJl-DH2hNXxKc0stVP8Qn56BNuTj5ytnismfADWvM8-Rxu8r20wJ7IELJCYE

Private Key:

e8PjoKxp3VPcxlxEp9XFG0D1rCiZy4FAtMuZ6ucN6ro



```
const webpush = require("web-push");

webpush.setVapidDetails(
  'mailto:julian2100.g.g@gmail.com',
  process.env.PUBLIC_KEY,
  process.env.PRIVATE_KEY
);
```

Aplicación desde cero: push notifications (server)



```
router.post('/', function(req, res, next) {  
  console.log(req.body);  
  const {endpoint, expirationTime, keys: {auth, p256dh}} = req.body;  
  Subscription.create({  
    endpoint,  
    expirationTime,  
    keys_auth: auth,  
    keys_p256dh: p256dh  
  });  
  res.sendStatus(200)  
});
```


Aplicación desde cero: push notifications (server)

```
router.post("/", async function (req, res, next) {  
  const data = req.body;  
  const news = await News.create(data);  
  await Subscription.findAll({}).then(function(subscriptions) {  
    subscriptions.forEach(function(subscription){  
      const {endpoint, expirationTime, keys_auth, keys_p256dh} = subscription;  
      real_subscription = {  
        endpoint,  
        expirationTime,  
        keys: {  
          auth: keys_auth,  
          p256dh: keys_p256dh  
        }  
      }  
    }  
    webpush.sendNotification(  
      real_subscription,  
      JSON.stringify(data)  
    ).catch(() => {});  
  })  
  res.send(news);  
})
```

Aplicación desde cero: push notifications (client)

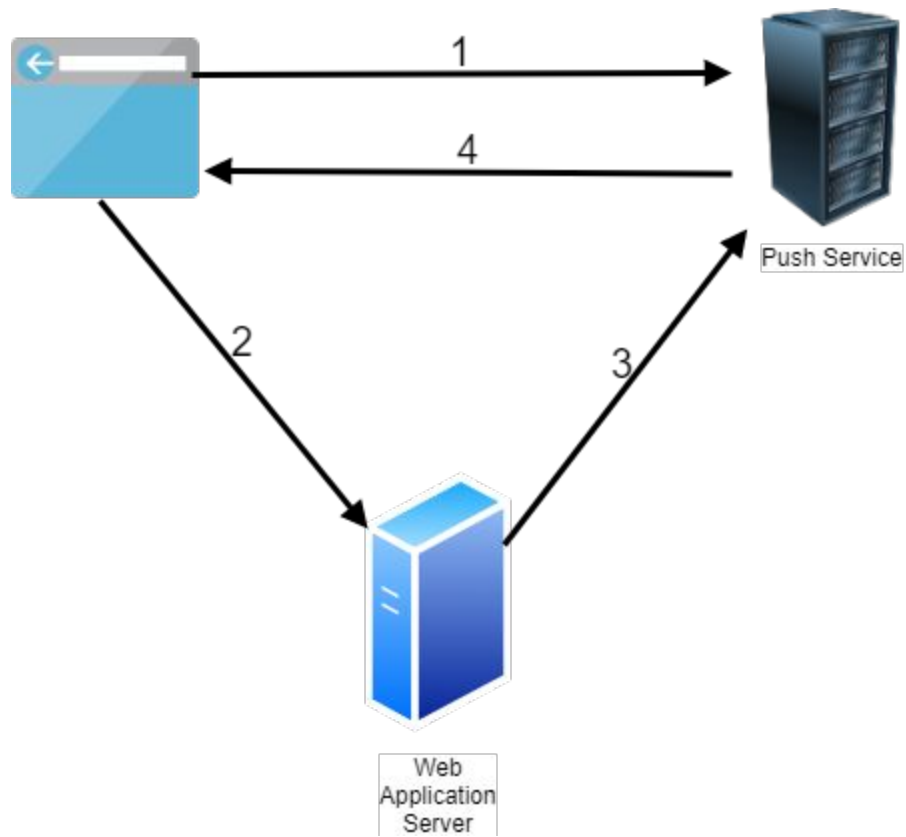
```
const configurePushSubscription = async () => {
  let serviceWorkerRegistration;
  navigator.serviceWorker.ready
    .then(registration => {
      serviceWorkerRegistration = registration;
      return registration.pushManager.getSubscription();
    })
    .then(subscription => {
      if (subscription === null) {
        return serviceWorkerRegistration.pushManager.subscribe({
          userVisibleOnly: true,
          applicationServerKey:
            'BAFYB80kg4DcyP-DL4_0vbk9gJY57qIroku12qjbYyytFJ1qm4hoa6EH6pmFCA7AqmaTK_w0xYPV-3A15z0qV-w'
        });
      }
      return subscription;
    })
    .then(pushSubscription => {
      return fetch("/subscriptions", {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
          'Accept': 'application/json'
        },
        body: JSON.stringify(pushSubscription)
      });
    })
    .catch(error => console.log(error));
};
```

Aplicación desde cero: push notifications (client)



```
self.addEventListener('push', function(e) {  
  const {title, body} = JSON.parse(e.data.text());  
  console.log(title, body);  
  e.waitUntil(  
    self.registration.showNotification(title, {body: body})  
  );  
});
```

Aplicación desde cero: push notifications



Aplicación desde cero: push notifications

```
{  
  endpoint: 'https://updates.push.services.mozilla.com/wpush  
/v2/gAAAAABieHYJyHbgqjtzqqjHU2TBgrccIe1Y5CkN6PmAkJdHArzweS6xyqiMB1oGFuaycJ8yVksHDLbxGZClbwB28hk3KNUnc  
QwsJPKICKyCdSbRlBBu6V3GPpg8P_lnbGgtNwYvzwcylvdSaB7YAtXirVZ6iPHVGL12qn4_SJqQlzsANX7FQ',  
  expirationTime: null,  
  keys: {  
    auth: 'bx_2WctaBYlBnqJPkidJVA',  
    p256dh: 'BKqV3-m-cAfdX7SHIuUV_2TkD72eIUJASmM1BiZ-mHoeHydect1sSInlTu71eQHBvi8vrCdPb3OSwFHEldA9ky4'  
  }  
}
```

Aplicación desde cero: push notifications

```
{
  endpoint: 'https://fcm.googleapis.com/fcm/send/eTG-
LuLspsw:APA91bHGEldvTIUA3Vm2P_11S7yliXWEUo8KgsB3Q7dLfMUaGLPt2WbN57FZkvB_uzz7yeJJYiXl0RCfe2e00v5Yf_RTEc
vYXbgszX5cb8oJe0XJ916rU2cJKLEoNc4juywLNM0yFz8N',
  expirationTime: null,
  keys: {
    p256dh: 'BATiRuaWCM0kRMUjd1RtGAh4UXEBKAQ1CjknPB7Qq7SFYkSerKoTW-8mjL1PhwYU0XY2YVpiI1CWBUz9Ktm_rMU',
    auth: 'Q_Ndtan5EuqIcVm4pfB-yw'
  }
}
```

Aplicación desde cero: IndexedDB

The screenshot shows a web application with two news items. The first item is titled "Julian lo Rompio?" with a "No" status. The second item is titled "Piñera lo pierde todo" with a "Cuerpo" status. Below the news items, the Chrome DevTools Storage Inspector is open, displaying a tree view of storage keys. The "owner" key is selected, and its value is shown in the right pane.

News items:

- Julian lo Rompio?
No
- Piñera lo pierde todo
Cuerpo

Storage Inspector:

- collectionParents
- documentMutations
- mutationQueues
- mutations
- owner**
- remoteDocumentChanges
- remoteDocumentGlobal
- remoteDocuments
- targetDocuments
- targetGlobal
- targets
- undefined (default)

Filter Items

Key	Value
owner	{"ownerId":"kE5vyPfAgk7CrbAtdHGW","allowTabSynchronization":false,"leaseTimestampMs":16...

05

Comparando implementaciones

Firestore

Ventajas:

- Out of the box
- Librería muy completa
- Sincronización automática entre la indexedDB y firestore

Desventajas:

- Side-effects son pagados

Desde cero

Ventajas:

- Más configurable
- Control más granular
- Side effects

Desventajas:

- Costoso de implementar (time-consuming)
- Muchos casos borde

06

Principales problemas

- Conexión de notificaciones con firebase.
- Sincronización entre IndexedDB y el servidor.
- Debugging de service workers es muy precario.
- Workarounds por falta de cloud functions.

07

Aprendizajes

- Firebase provee herramientas muy útiles para desarrollar una pwa, aunque algunas funcionalidades son pagadas.
- Desarrollar una librería de herramientas para pwa puede ser muy costoso.
- Los service workers están en una etapa muy precaria aún.
- Una pwa es una muy buena alternativa a desarrollar aplicaciones nativas, ya que se ahorra tiempo de desarrollo y se puede llegar a parecer mucho a una.