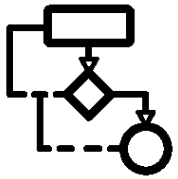
Web Capabilites

El nuevo futuro para la web

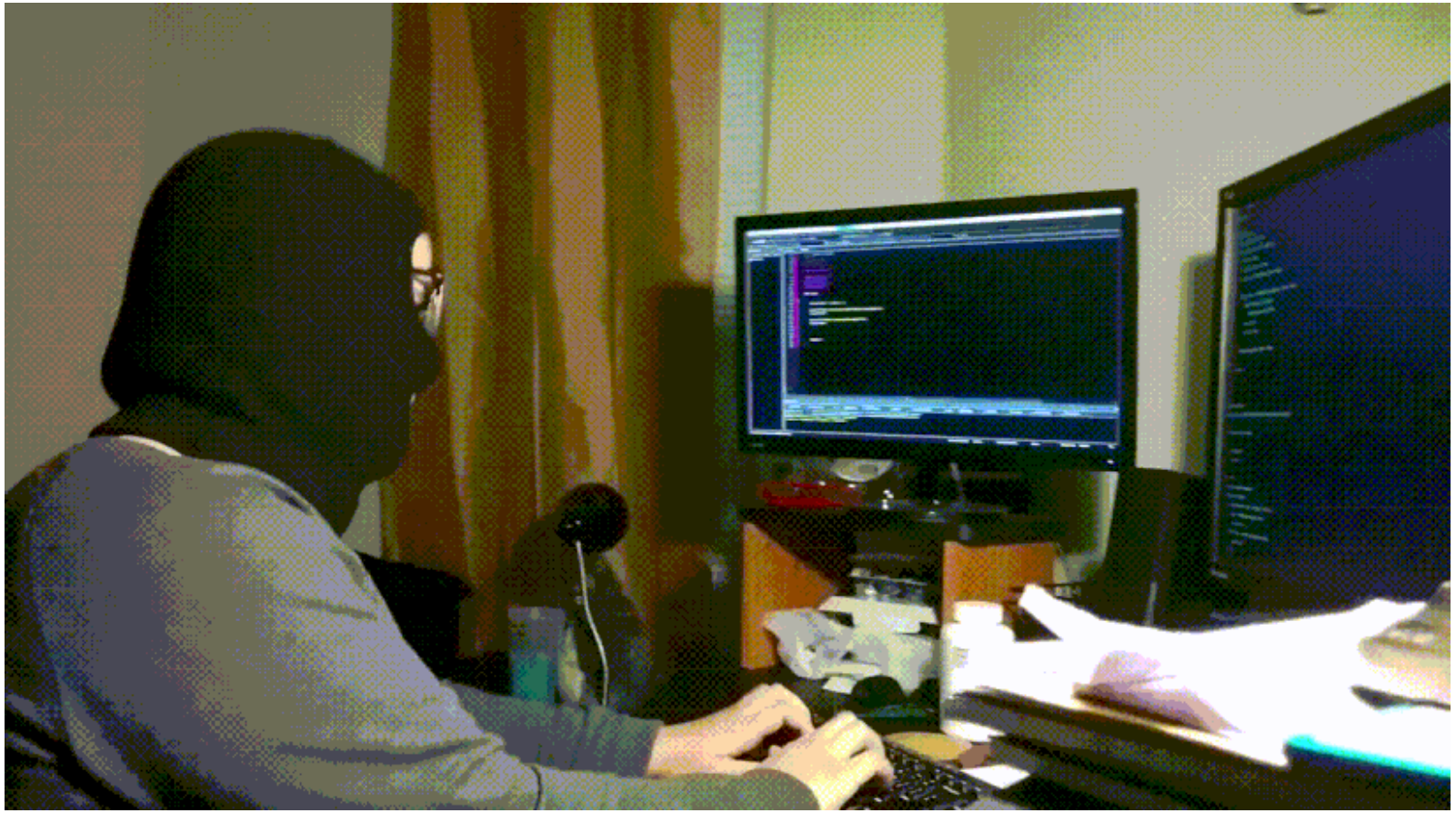
Alex González



https://github.com/agonsant

https://www.linkedin.com/in/alejandro-gonzalez-santiago/

Expectativa



Realidad

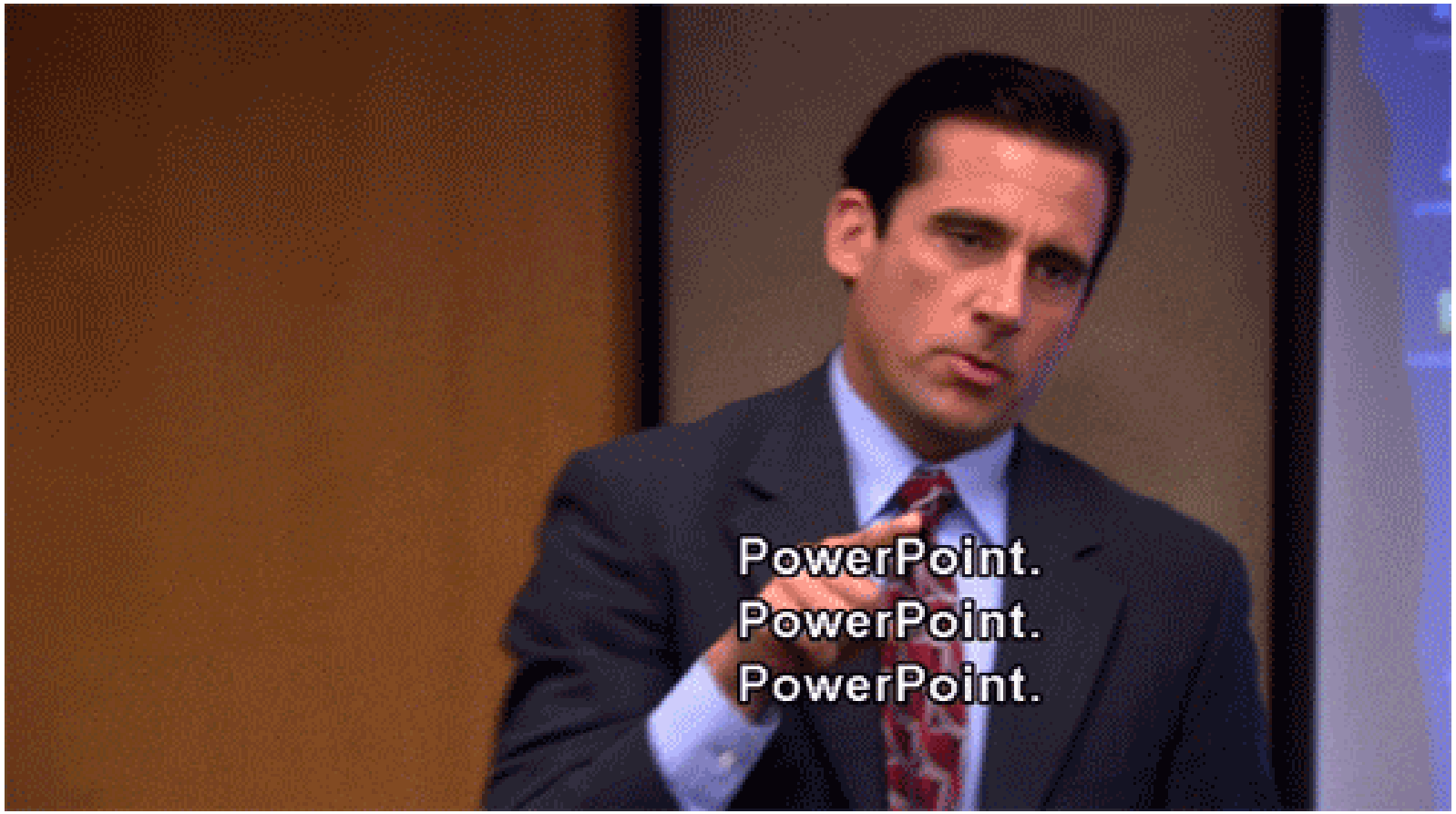


Image from problogooster.com


Android
Web
IOS


Tenemos que sacar nuestro canal online en Android, IOS y Web responsive
Nativo
Nativo
¿Y si hacemos una app híbrida?
imgflip.com

Así solo escribimos un código

Es más barato

Time To Market

Así solo escribimos un código

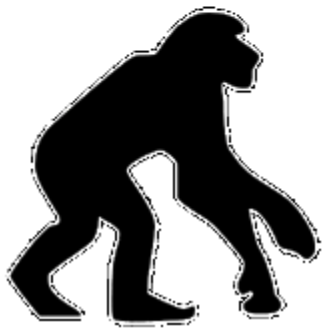
Es más barato

Time To Market

Las híbridas van lentas

Mal UX

Nativo

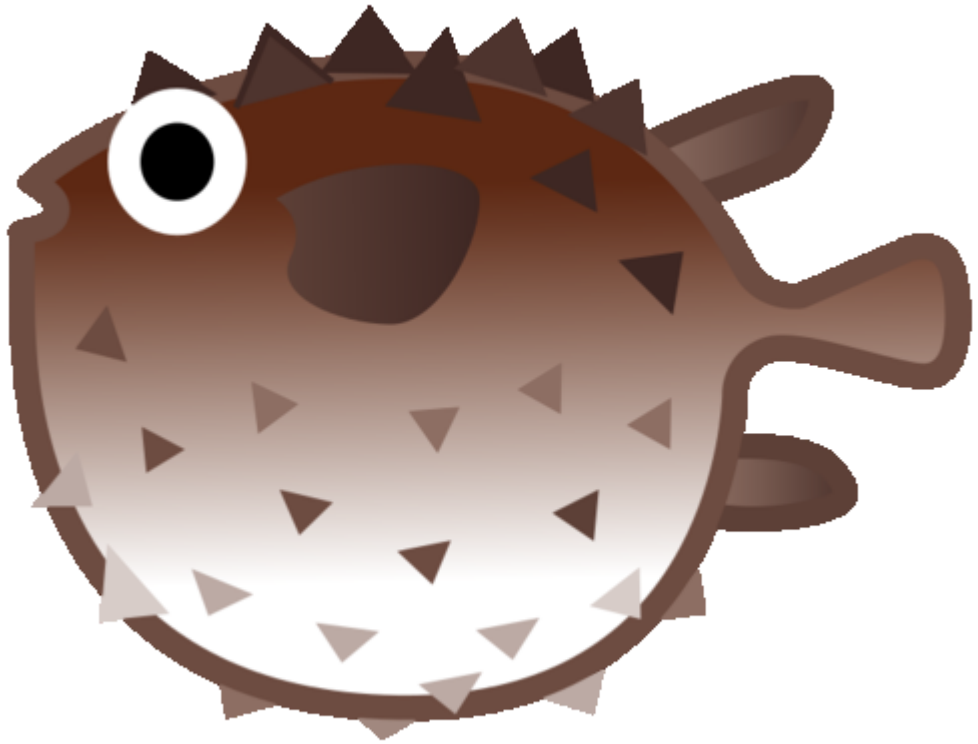


PhoneGap /
Apache Cordova

Xamarin, React
Native, Ionic,
Capacitor …

Progressive Web Apps

Web Capabilities (Project Fugu)



https://codelabs.developers.google.com/dev-pwa-training/

Bluetooth
Offline Mode
Device Memory
Battery Status
Device position
Speech Recognition
Facial Recognition
Real-Time Communications
Screen orientation & Lock
Credentials & Authentication
Contacts
Share
USB

Badging
NFC
Advance Camera Controls
Native File System
Background Sync
Recording Media
Serial Devices
Push Messages
Geolocation
Payments
SMS OTP

https://whatwebcando.today/

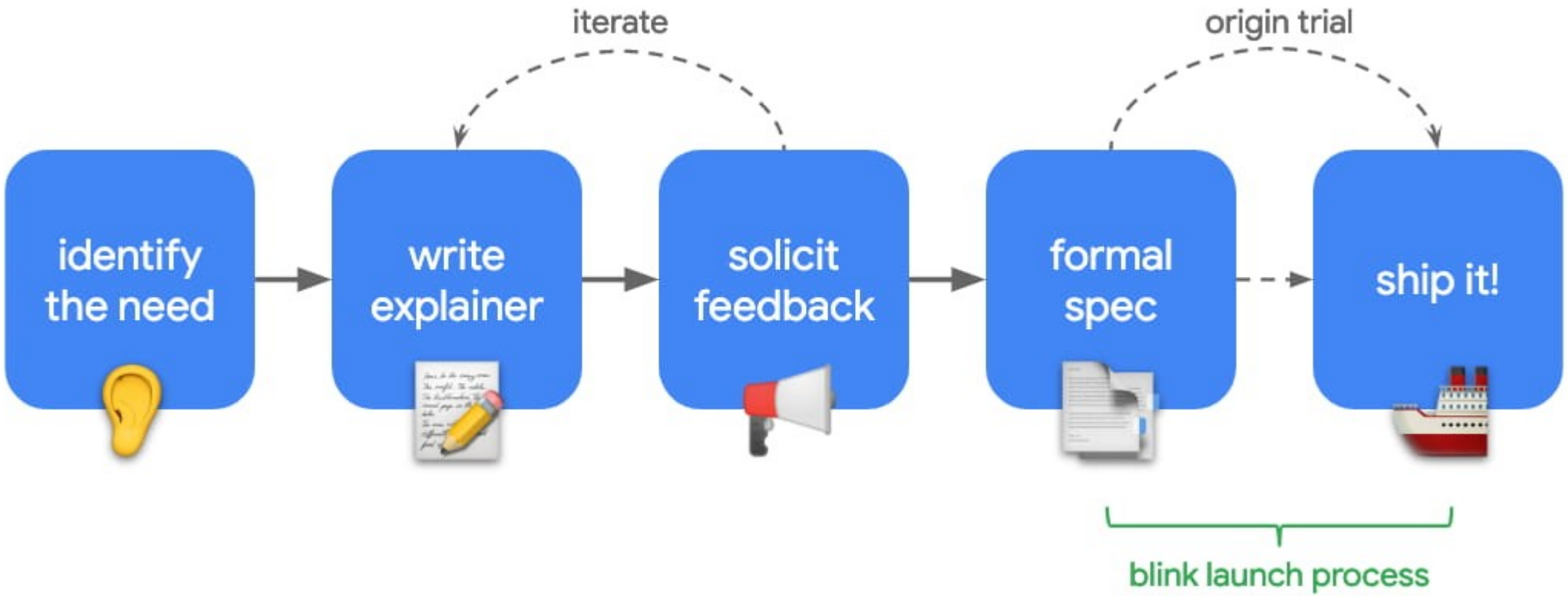
Fugu Landing Page

Fugu New Capabilities Status

Web Capabilities Codelabs

Unlocking New Capabilities for the Web (Google I/O ’19)

Image from [developers.google.com]



[New Feature Request]

[Test new Trial API's]

Web Share API

Nos permite compartir Texto, URL o un archivo (A partir de la V2) utilizando el sistema nativo de compartición

Tiene las siguientes limitaciones por seguridad:

- Solo se puede invocar en páginas cargadas por HTTPS
- No se puede invocar sin una interacción previa del usuario en la página como un click.

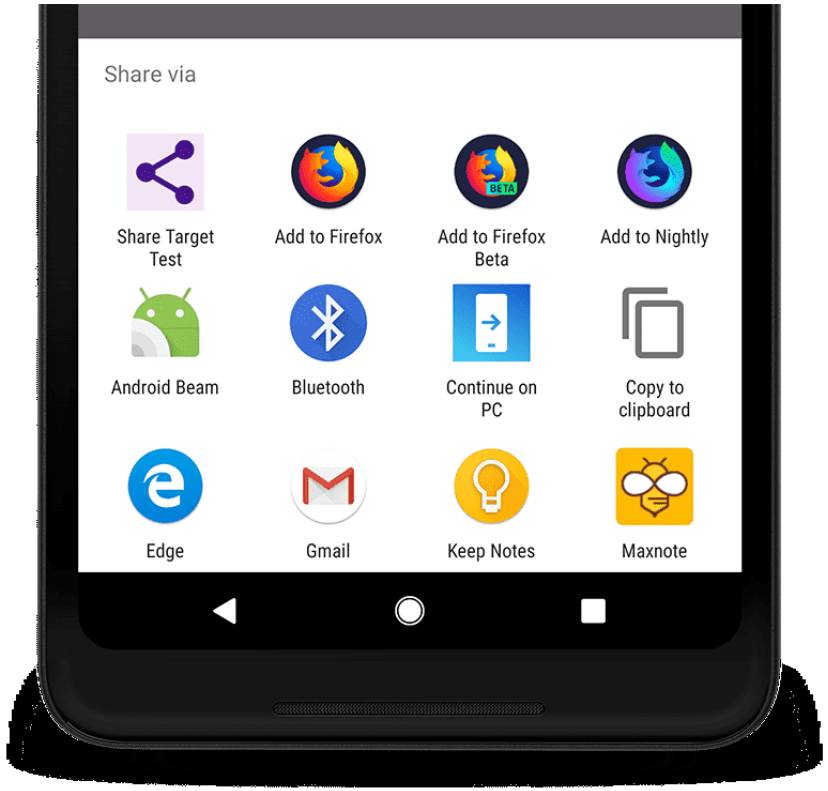


Image from web.dev

https://web.dev/web-share/

https://w3c.github.io/web-share/demos/share-files.html

Web Share API

```
if (navigator.share) {
    //At least one of these parameters is required
    navigator.share({
        title: 'The Share Title',
        text: 'A Sharable Text',
        url: 'https://sharable.url/',
    })
    .then(() => console.log('Successful share'))
    .catch((error) => console.log('Error sharing', error));
}
```

V2

```
// filesArray is a FileList object (similar to NodeList)
const filesArray = document.getElementById('myfileinput').files;

if (navigator.canShare && navigator.canShare({ files: filesArray })) {
    navigator.share({
        files: filesArray,
        title: 'File Share Titles',
        text: 'Files for a meeting',
    })
    .then(() => console.log('Share was successful.'))
    .catch((error) => console.log('Sharing failed', error));
} else {
    console.log(`Your system doesn't support sharing files.`);
}
```

Web Share API

share	✗	✗	✗	✓	✓	✓	✓
canShare	✗	✗	✗	✗	✓	✗	✗

Can I Use

Web Share Target API

Nos permite registrar nuestra PWA instalada como aplicación a la que puedes compartir registrándola directamente en el SO

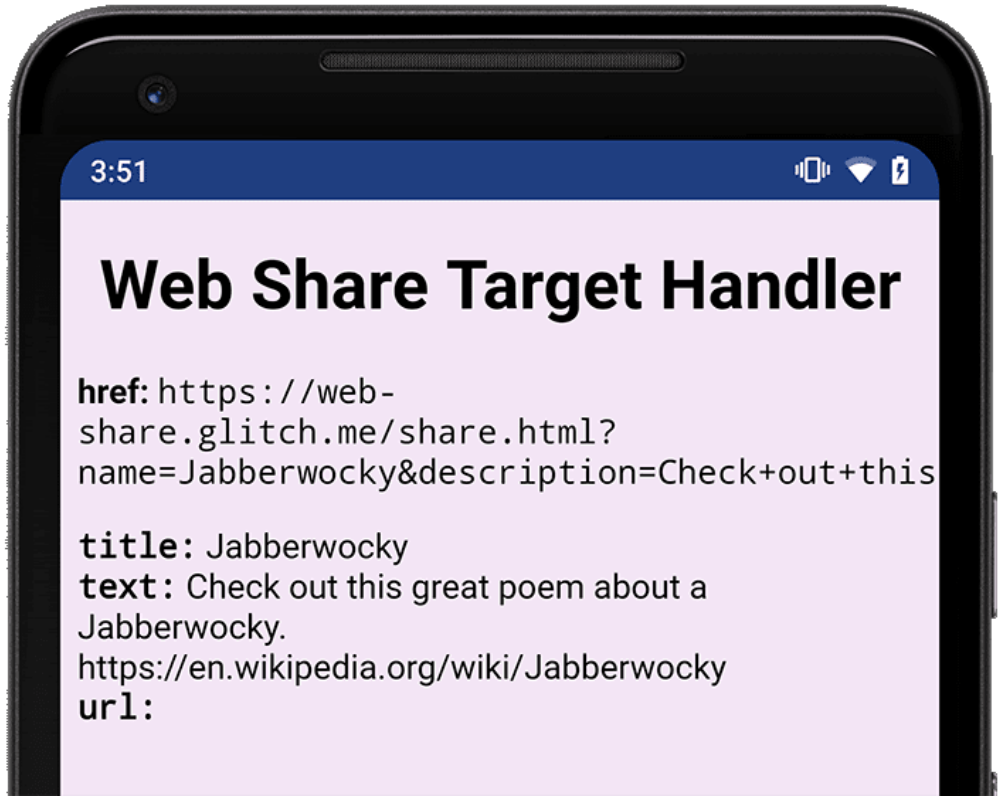


Image from web.dev

https://web.dev/web-share-target/

https://wicg.github.io/web-share-target/demos/sharetarget.html

https://agonsant.github.io/roll-2-roll/

Web Share Target API. Manifest Registration

Basic

```
"share_target": {
    "action": "/my-share-target-page/",
    "method": "GET",
    "params": {
        "title": "title",
        "text": "text",
        "url": "url"
    }
}
```

Basic POST

```
"share_target": {
  "action": "/update-something-page",
  "method": "POST",
  "enctype": "multipart/form-data",
  "params": {
    "url": "link"
  }
}
```

POST with Files

```
"share_target": {
    "action": "/my-shared-post-page",
    "method": "POST",
    "enctype": "multipart/form-data",
    "params": {
        "title": "name",
        "text": "description",
        "url": "link",
        "files": [
            {
                "name": "records",
                "accept": [
                    "text/csv",
                    ".csv"
                ]
            },
            {
                "name": "graphs",
                "accept": "image/svg+xml"
            }
        ]
    }
}
```

Web Share Target API. Handle incoming

```
window.addEventListener('DOMContentLoaded', () => {
    const parsedUrl = new URL(window.location);
    // searchParams.get() will properly handle decoding the values.
    console.log('Title shared: ' + parsedUrl.searchParams.get('title'));
    console.log('Text shared: ' + parsedUrl.searchParams.get('text'));
    console.log('URL shared: ' + parsedUrl.searchParams.get('url'));
});
```

Web Share Target API. Handle incoming

```
/**
 * As it is a POST with multipart/form-data encypt, the page can not process it directly.
 * We have to process it in our Service Worker and pass it to the page using postMessage()
 * or directly to the server *
 * */
self.addEventListener('fetch', event => {
    if (event.request.method !== 'POST') {
        event.respondWith(fetch(event.request));
        return;
    }

    event.respondWith((async () => {
        const formData = await event.request.formData();
        const link = formData.get('link') || '';
        const responseUrl = await saveLink(link);
        return Response.redirect(responseUrl, 303);
    })());
});
```

Web Share API

share_target

Native File System API

chrome://flags -> #native-file-system-api

Nos permite acceder directamente sistema de ficheros nativo del dispositivo, para lectura y escritura.

Tiene las siguientes limitaciones por seguridad:

- No se puede acceder a carpetas restringidas o propias de las librerías del SO
- Para poder escribir se tendrá que solicitar el permiso al usuario.

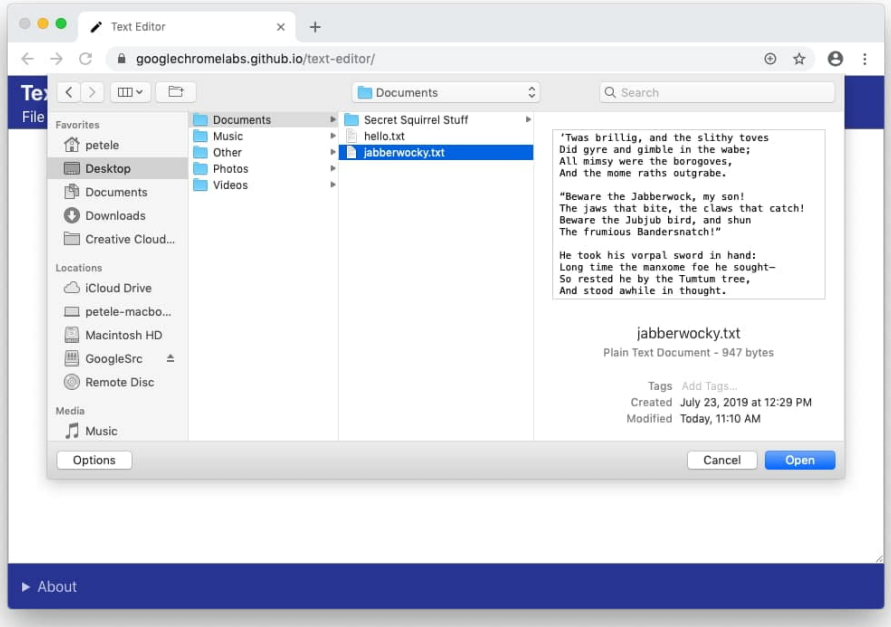


Image from web.dev

https://web.dev/native-file-system

https://googlechromelabs.github.io/text-editor/

Native File System API. Reader

```
let fileHandle;
butOpenFile.addEventListener('click', async (e) => {
  fileHandle = await window.chooseFileSystemEntries();
  const file = await fileHandle.getFile();
  const contents = await file.text();
  textArea.value = contents;
});
```

Native File System API. Writer

```
async function getNewFileHandle() {
    const opts = {
        type: 'saveFile',
        accepts: [{
            description: 'Text file',
            extensions: ['txt'],
            mimeTypes: ['text/plain'],
        }],
    };
    const handle = await window.chooseFileSystemEntries(opts);
    return handle;
}
```

```
async function writeFile(fileHandle, contents) {
    // Create a writer (request permission if necessary).
    const writer = await fileHandle.createWriter();
    // Write the full length of the contents
    await writer.write(0, contents);
    // Close the file and write the contents to disk
    await writer.close();
}
```

Native File System API. Directory

```
const butDir = document.getElementById('butDirectory');
butDir.addEventListener('click', async (e) => {
    const opts = { type: 'openDirectory' };
    const handle = await window.chooseFileSystemEntries(opts);
    const entries = await handle.getEntries();
    for await (const entry of entries) {
        const kind = entry.isFile ? 'File' : 'Directory';
        console.log(kind, entry.name);
    }
});
```

Native File System API

chooseFileSystemEntries

Badging API

chrome://flags -> #enable-experimental-web-platform-features

Nos permite utilizar la notificación de nueva actividad de manera nativa para una PWA instalada.

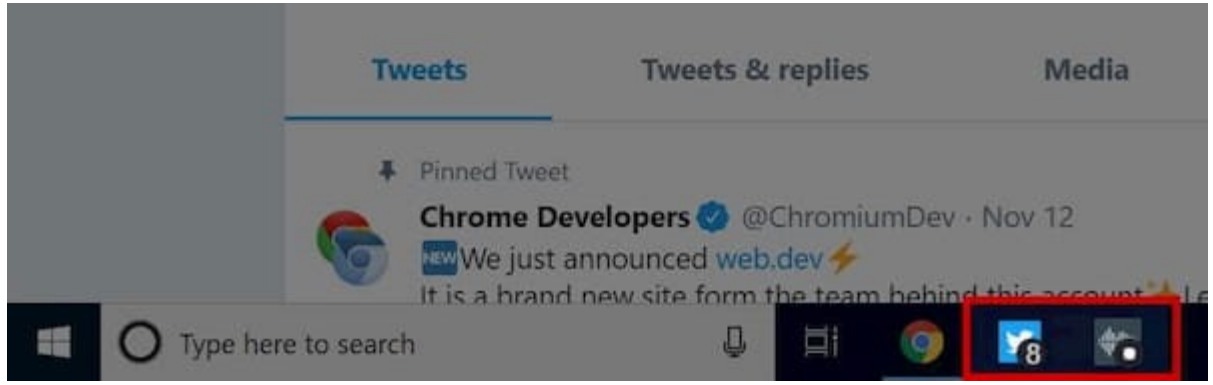


Image from web.dev

https://web.dev/badging-api/

https://badging-api.glitch.me/

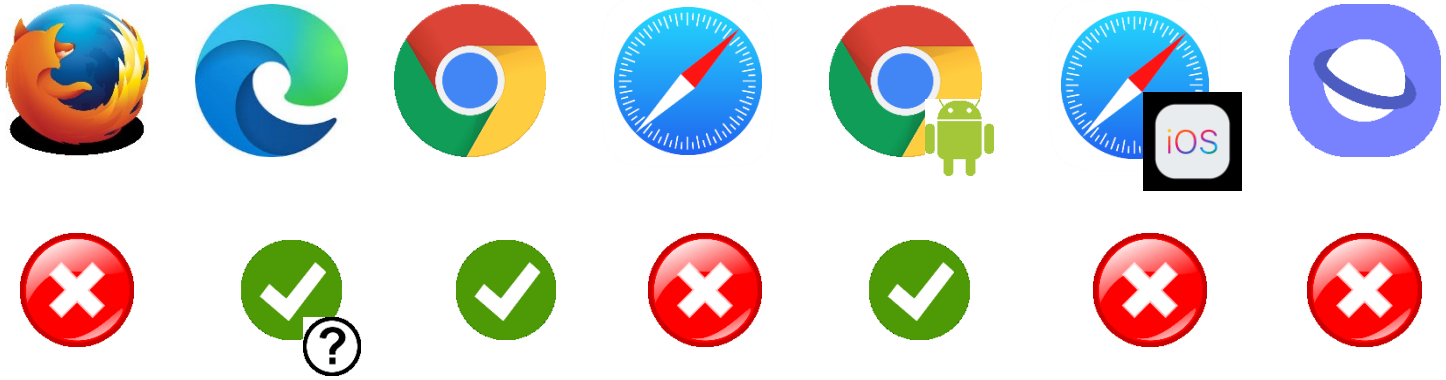
Badging API

```
// Set the badge
const unreadCount = 2;
navigator.setBadge(unreadCount);

// Clear the badge
navigator.clearBadge();
```

Badging API

setBadge / clearBadge



Contact Picker API

Nos permite acceder a los contactos del dispositivo y poder seleccionarlos.

Tiene las siguientes limitaciones por seguridad:

- Para poder escribir se tendrá que solicitar el permiso al usuario y este tendrá que seleccionar los contactos a compartir

https://web.dev/contact-picker/

https://contact-picker.glitch.me/

Video from web.dev

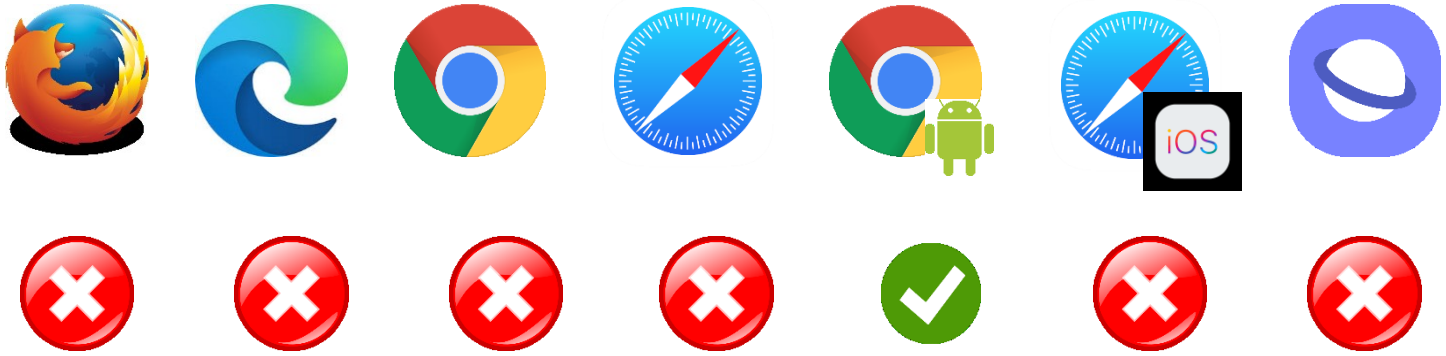
Contact Picker API

```
if ('contacts' in navigator && 'ContactsManager' in window) {
    const props = ['name', 'email', 'tel', 'address', 'icon'];
    const opts = { multiple: true };

    try {
        const contacts = await navigator.contacts.select(props, opts);
        /**
         * Contacts === [{
                            "email": [],
                            "name": ["Queen O'Hearts"],
                            "tel": ["+1-206-555-1000", "+1-206-555-1111"]
                        }]
         */
        handleResults(contacts);
    } catch (ex) {
        // Handle any errors here.
    }
} else {
    // CONTACT PICKER API NOT SUPPORTED
}
```

Contact Picker API

contacts /
Contacts Manager



Wake Lock API

chrome://flags -> #enable-experimental-web-platform-features

Evita que el dispositivo se “duerma” cuando se encuentra la app en IDLE, no permitiendo que se apague la pantalla

El API provee dos tipos de bloqueos, *screen* y *system(todavía no implementada):*

- *screen* => Evita que se apague la pantalla del dispositivo
- *system* => Evita que la CPU se ponga en stanby para que la app pueda seguir ejecutándose.

Este API impacta directamente en los recursos del sistema, hay que gestionarla correctamente para evitar drenado rápido de la batería al usar nuestra APP

https://web.dev/wakelock/

https://wake-lock-demo.glitch.me/

Wake Lock API

```
// The wake lock sentinel.
let wakeLock = null;

// Function that attempts to request a wake lock.
const requestWakeLock = async () => {
    try {
        wakeLock = await navigator.wakeLock.request('screen');
        wakeLock.addEventListener('release', () => {
            console.log('Wake Lock was released');
        });
        console.log('Wake Lock is active');
    } catch (err) {
        console.error(`${err.name}, ${err.message}`);
    }
};

// Request a wake lock…
await requestWakeLock();
// …and release it again after 5s.
window.setTimeout(() => {
    wakeLock.release();
    wakeLock = null;
}, 5000);

// Listen to the wake lock lifecycle
const handleVisibilityChange = () => {
    if (wakeLock !== null && document.visibilityState === 'visible') {
        requestWakeLock();
    }
};

document.addEventListener('visibilitychange', handleVisibilityChange);
document.addEventListener('fullscreenchange', handleVisibilityChange);
```

Wake Lock API

							
Wake Lock							

Web Notifications API

chrome://flags -> #enable-experimental-web-platform-features

Desde hace tiempo las Web Apps pueden enviar notificaciones al usuario.

Con la extensión de este API se permite programar una notificación a un usuario para que salte en un tiempo determinado, incluso estando offline. Se suele usar conjunto con el Push API para combinar todo el sistema de notificaciones en una App.

https://web.dev/notification-triggers/

https://notification-triggers.glitch.me/

Web Notifications API

```
const createScheduledNotification = async (tag, title, timestamp) => {
    const registration = await navigator.serviceWorker.getRegistration();
    registration.showNotification(title, {
        tag: tag,
        body: "This notification was scheduled 30 seconds ago",
        showTrigger: new TimestampTrigger(timestamp + 30 * 1000)
    });
};


const cancelScheduledNotification = async (tag) => {
    const registration = await navigator.serviceWorker.getRegistration();
    const notifications = await registration.getNotifications({
        tag: tag,
        includeTriggered: true,
    });
    notifications.forEach((notification) => notification.close());
};
```

Web Notifications API

Notifications	✓	✓	✓	✓	✓	✓	✓
showTrigger/ includeTriggered	✗	✓?	✓	✗	✓	✗	✗

Web Bluetooth API

chrome://flags -> #enable-experimental-web-platform-features

Nos permite conectar y comunicarnos con dispositivos Bluetooth que tengamos a nuestro alrededor.

Tiene las siguientes limitaciones por seguridad:

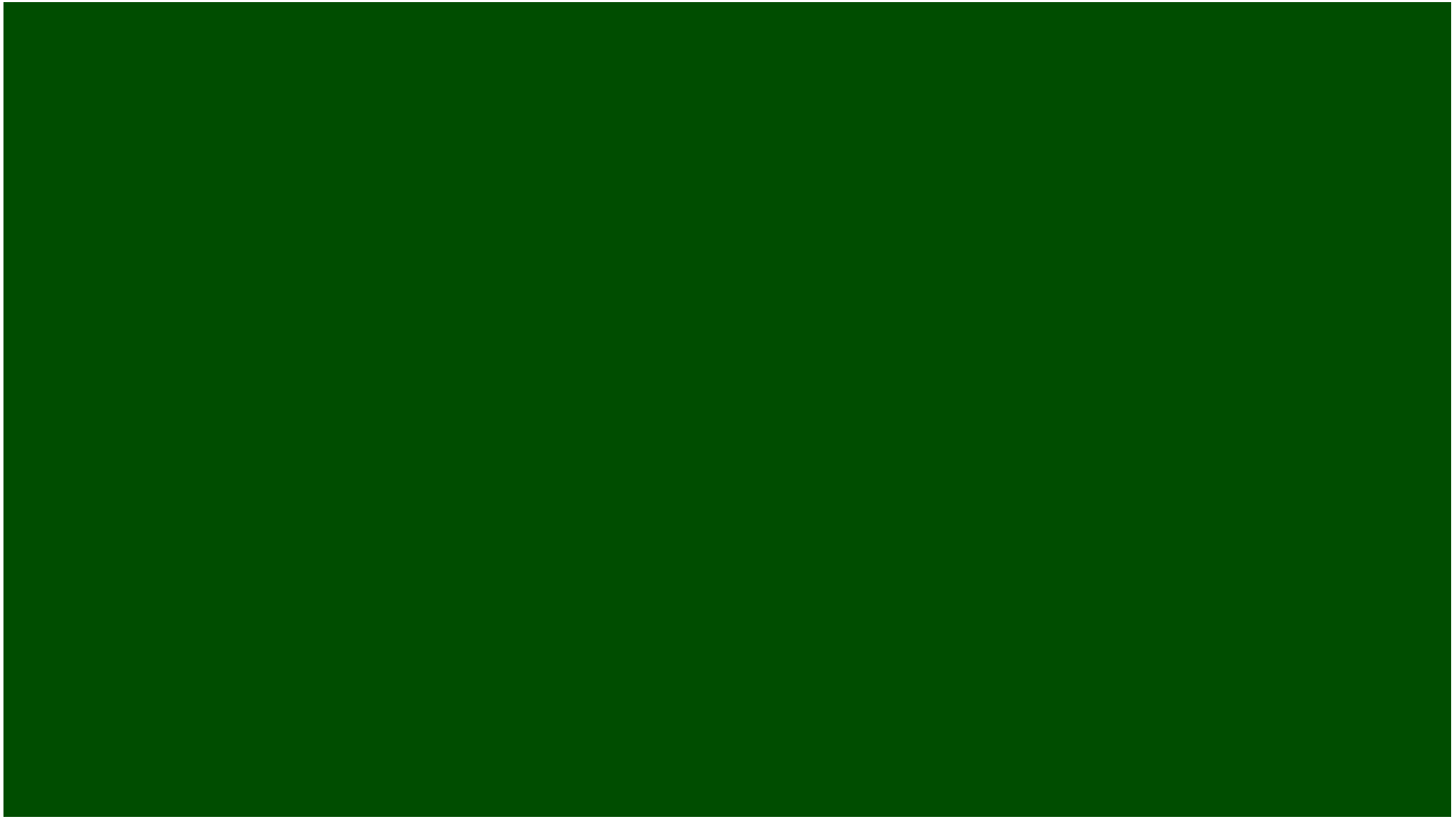
- Actualmente el API solo permite conectarse con ***dispositivos GATT (Generic Attribute Profile) sobre BLE (Bluetooth Low Energy) y que además implementan Bluetooth 4.0 o superior.***
- Solo se puede utilizar sobre HTTPS
- Es necesario una interacción del usuario para conectarse.

Para depuración y desarrollo podemos ver los dispositivos que tenemos alrededor en *chrome://bluetooth-internals*

https://webbluetoothcg.github.io/web-bluetooth/

https://googlechrome.github.io/samples/web-bluetooth/

Web Bluetooth API



Video from developers.google.com

Web Bluetooth API

```
/**
 * STEP 1: REQUEST AND SCAN DEVICES
 */
const device = await navigator.bluetooth.requestDevice({ filters: [{ services: ['battery_service'] }] });

/**
 * STEP 2: CONNECT TO THE GATT SERVER
 */
const server = await device.gatt.connect();

/**
 * STEP 3: GET THE SERVICE FROM GATT SERVER
 */
const service = await server.getPrimaryService('battery_service');

/**
 * STEP 4: GET THE CHARACTERISTIC FROM SERVICE
 */
const characteristic = await service.getCharacteristic('battery_level');

/**
 * STEP 5: HANDLE CHARACTERISTIC VALUE CHANGE
 */
function handleBatteryLevelChanged(event) {
    let batteryLevel = event.target.value.getUint8(0);
    console.log('Battery percentage is ' + batteryLevel);
}
characteristic.addEventListener('characteristicvaluechanged', handleBatteryLevelChanged);

/**
 * STEP 6: READ THE CHARACTERISTIC VALUE
 */
const value = await characteristic.readValue();

console.log('Battery percentage is ' + value.getUint8(0));
```

Web Bluetooth API

Web Bluetooth API

More APIs…

- NFC => Conexión con dispositivos NFC. De momento limitado a NDEF. Demos
- USB => Conexión con periféricos USB
- Media Session API => Permite integrarse con el reproductor media del dispositivo Media Session API
- Shape Detection API => Api para la detección Facial, QR, Barcodes, OCR. Demo
- Serial API => Comunicar una Web con Serial Devices. Demo
- WebHID API => Conexión con los Human Interface Devices. Status
- SMS API OTP => Ayuda al acceso del OTP por SMS sin que el usuario tenga que salir de la app.
- Local Font Access API, Background Sync API, Content Indexing API, Payment API, Credential API, Web Authentication API, …