mozilla

#FirefoxOSAppDays

mozilla

WEBAPI

Llevando el poder a las tecnologías web

#FirefoxOSAppDays

mozilla

GUILLERMO LÓPEZ

@willyaranda

#FirefoxOSAppDays

mozilla

Todo HTML, CSS y JavaScript

#FirefoxOSAppDays

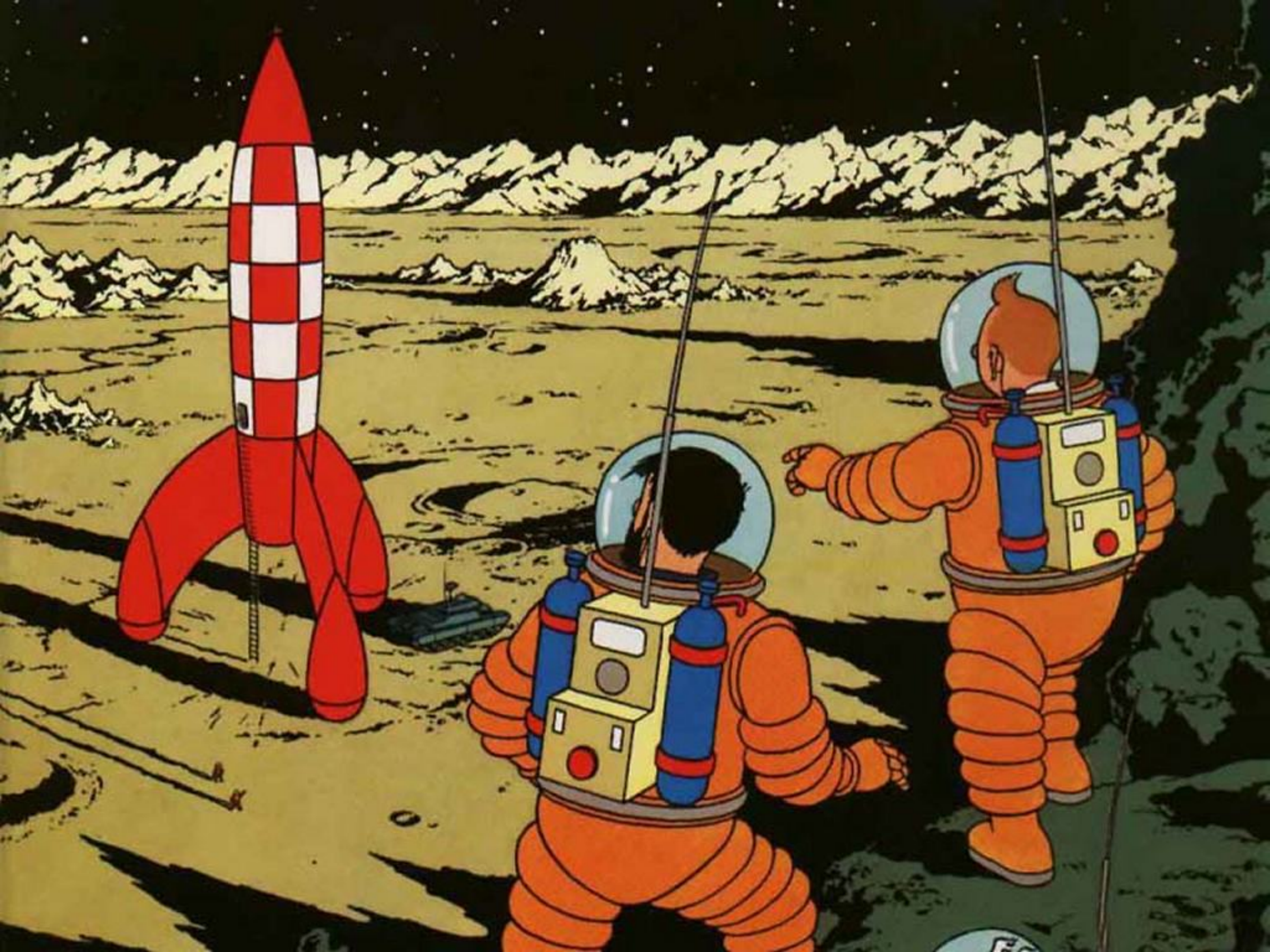
El equipo de WebAPI de Mozilla está empujando la web hacia sus límites para incluir, y en algunos casos superar, las posibilidades de otras plataformas

#FirefoxOSAppDays

Usuarios

Desarrolladores

Navegador
(APIs)



Niveles de seguridad

#FirefoxOSAppDays

mozilla

Por defecto

WebGL, pantalla completa, audio

Aceptadas por el usuario:

Ubicación, cámara, acceso a tarjeta de memoria

Al instalar:

IndexedDB, offline caché, localStorage (sin límite)

Aceptadas por la tienda:

APIs de privacidad: Contactos...

Verificadas

Alto privilegio: radio (telefonía y SMS)

#FirefoxOSAppDays

Contenido web

Aplicación web instalada

Aplicación web privilegiada

Aplicación web certificada

```
"permissions": {
    "contacts": {
        "description": "Required for autocompletion in the share screen",
        "access": "readcreate"
    },
    "alarms": {
        "description": "Required to schedule notifications"
    }
}
```

Manifest.webapp

AlarmAPI
BrowserAPI
Contacts
Storage
FMRadio
Geolocation
SystemXHR
TCP Socket API
Wake-lock

Necesitan permiso

mozilla

VibrationAPI (W3C)
Screen Orientation
Geolocation API (W3C)
Mouse Lock API (W3C)
Open WebApps
Network Information API (W3C)
Battery Status API (W3C)
Alarm API
Web Activities
Push Notifications API
WebFM API
WebPayment
IndexedDB (W3C)
Ambient Light Sensor
Proximity Sensor
Notifications API

BARRA LIBRE

#FirefoxOSAppDays

```
var battery = navigator.battery;
if (battery) {
        var batteryLevel = Math.round(battery.level * 100) + "%",
            charging = (battery.charging)? "" : "not ",
            chargingTime = parseInt(battery.chargingTime / 60, 10,
            dischargingTime = parseInt(battery.dischargingTime / 60, 10);

    // Set events
    battery.addEventListener("levelchange", setStatus, false);
    battery.addEventListener("chargingchange", setStatus, false);
    battery.addEventListener("chargingtimechange", setStatus, false);
    battery.addEventListener("dischargingtimechange", setStatus, false);
}
```

Batería

```
var notification = navigator.mozNotification;
notification.createNotification(
    "See this",
    "This is a notification",
    iconURL
);
```

Notificaciones

```
// Portrait mode:
screen.mozLockOrientation("portrait");

/*
    Possible values:
        "landscape"
        "portrait"
        "landscape-primary"
        "landscape-secondary"
        "portrait-primary"
        "portrait-secondary"
*/
```

Orientación

```
// Vibrate for one second
navigator.vibrate(1000);

// Vibration pattern [vibrationTime, pause,…]
navigator.vibrate([200, 100, 200, 100]);

// Vibrate for 5 seconds
navigator.vibrate(5000);

// Turn off vibration
navigator.vibrate(0);
```

Vibración

```
var pay = navigator.mozPay(paymentToken);
pay.onsuccess = function (event) {
    // Weee! Money!
};
```

WebPayments

mozilla

```
var connection = window.navigator.mozConnection,
    online = connection.bandwidth > 0,
    metered = connection.metered;
```

Network Information API

#FirefoxOSAppDays

```
var alarmId1,
    request = navigator.mozAlarms.add(
        new Date("May 15, 2012 16:20:00"),
        "honorTimezone",
        {
            mydata: "my event"
        }
    );

request.onsuccess = function (event) {
    alarmId1 = event.target.result;
};

request.onerror = function (event) {
    console.log(event.target.error.name);
};
```

AlarmAPI

```
var request = navigator.mozAlarms.getAll();

request.onsuccess = function (event) {
    console.log(JSON.stringify(event.target.result));
};

request.onerror = function (event) {
    console.log(event.target.error.name);
};
```

```
navigator.mozAlarms.remove(alarmId1);
```

AlarmAPI

```
navigator.mozSetMessageHandler(
    "alarm",
    function (message) {
        // Note: message has to be set in the manifest file
        console.log("Alarm fired: " + JSON.stringify(message));
    }
);
```

```
{
    "messages": ["alarm"]
}
```

AlarmAPI

```
window.addEventListener("deviceproximity", function (event) {
    // Current device proximity, in centimeters
    console.log(event.value);

    // The maximum sensing distance the sensor is
    // able to report, in centimeters
    console.log(event.max);

    // The minimum sensing distance the sensor is
    // able to report, in centimeters
    console.log(event.min);
});
```

Proximidad

```
window.addEventListener("devicelight", function (event) {
    // The level of the ambient light in lux
    console.log(event.value);
});
```

```
window.addEventListener("lightlevel", function (event) {
    // Possible values: "normal", "bright", "dim"
    console.log(event.value);
});
```

```
window.addEventListener("devicelight", function (event) {
    //  The lux values for "dim" typically begin below 50,
    // and the values for "bright" begin above 10000
    console.log(event.value);
});
```

Intensidad lumínica

APIs privilegiadas

#FirefoxOSAppDays

```
var deviceStorage = navigator.getDeviceStorage("videos");
```

```
// "external", "shared", or "default".
deviceStorage.type;

// Add a file - returns DOMRequest with file name
deviceStorage.add(blob);

// Same as .add, with provided name
deviceStorage.addNamed(blob, name);

// Returns DOMRequest/non-editable File object
deviceStorage.get(name);

// Returns editable FileHandle object
deviceStorage.getEditable(name);

// Returns DOMRequest with success or failure
deviceStorage.delete(name);

// Enumerates files
deviceStorage.enumerate([directory]);

// Enumerates files as FileHandles
deviceStorage.enumerateEditable([directory]);
```

Device Storage

```
var storage = navigator.getDeviceStorage("videos"),
    cursor = storage.enumerate();

cursor.onerror = function() {
  console.error("Error in DeviceStorage.enumerate()", cursor.error.name);
};

cursor.onsuccess = function() {
    if (!cursor.result)
        return;

    var file = cursor.result;

    // If this isn't a video, skip it
    if (file.type.substring(0, 6) !== "video/") {
        cursor.continue();
        return;
    }

    // If it isn't playable, skip it
    var testplayer = document.createElement("video");
    if (!testplayer.canPlayType(file.type)) {
        cursor.continue();
        return;
    }
};
```

Device Storage

```
var deviceStorage = navigator.getDeviceStorage("videos");
```

```
// "external", "shared", or "default".
deviceStorage.type;

// Add a file - returns DOMRequest with file name
deviceStorage.add(blob);

// Same as .add, with provided name
deviceStorage.addNamed(blob, name);

// Returns DOMRequest/non-editable File object
deviceStorage.get(name);

// Returns editable FileHandle object
deviceStorage.getEditable(name);

// Returns DOMRequest with success or failure
deviceStorage.delete(name);

// Enumerates files
deviceStorage.enumerate([directory]);

// Enumerates files as FileHandles
deviceStorage.enumerateEditable([directory]);
```

Device Storage

```
var contact = new mozContact();
contact.init({name: "Tom"});

var request = navigator.mozContacts.save(contact);
request.onsuccess = function() {
    console.log("Success");
};

request.onerror = function() {
    console.log("Error")
};
```

Contactos

WebTelephony
WebSMS
Idle API
Settings API
Power Management API
Mobile Connection API
WiFi Information API
WebBluetooth
Permissions
Network Stats API
Camera API
Time/Clock API
Attention Screen
Voicemail

APIs Certificadas

```
// Telephony object
var tel = navigator.mozTelephony;

// Check if the phone is muted (read/write property)
console.log(tel.muted);

// Check if the speaker is enabled (read/write property)
console.log(tel.speakerEnabled);
```

```
// Place a call
var cal = tel.dial("123456789");
```

Telefonía

```
// SMS object
var sms = navigator.mozSMS;

// Send a message
sms.send("123456789", "Hello world!");
```

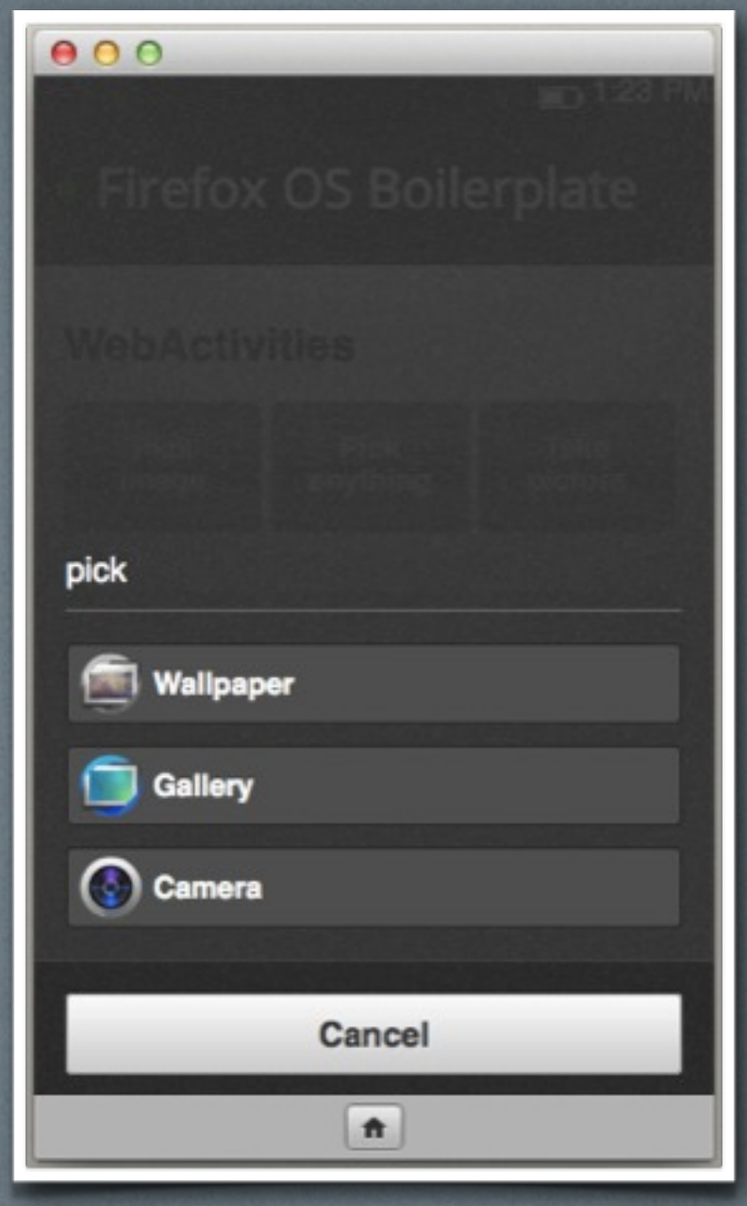
```
// Recieve a message
sms.onreceived = function (event) {
    // Read message
    console.log(event.message);
};
```

Telefonía

mozilla

WebActivities

#FirefoxOSAppDays



#FirefoxOSAppDays

```
{
    "activities": {
        "share": {
            "filters": {
                type: ["image/png", "image/gif"],
            }
            "href": "sharing.html",
            "disposition": "window"
        }
    }
}
```

Web Activities

```
var activity = new MozActivity({
    name: "view",
    data: {
        type: "image/png",
        url: ...
    }
});

activity.onsuccess = function () {
    console.log("Showing the image!");
};

activity.onerror = function () {
    console.log("Can't view the image!");
};
```

```
var register = navigator.mozRegisterActivityHandler({
    name: "view",
    disposition: "inline",
    filters: {
        type: "image/png"
    }
});

register.onerror = function () {
    console.log("Failed to register activity");
}
```

```
navigator.mozSetMessageHandler("activity", function (a) {
    var img = getImageObject();
    img.src = a.source.url;
    // Call a.postResult() or a.postError() if
    // the activity should return a value
});
```

Web Activities

WebAPI

DRAFT
This page is not complete.

WebAPIs enable acess to hardware of a device (like camera, battery or vibration) and to data which is stored or available on a device (like calendar or
WebAPI effort expands what the Web can do today and only proprietary platforms were able to do in the past.

DOCUMENTATION ABOUT WEBAPI

Documentation status

APIs implemented and drafted in a W3C specification

Battery API

Provides information about the system's battery charge level.

Network information API

Provides information about the system's connection, such as the current bandwidth.

Vibration API

Pulses the vibration hardware on the device.

Implemented but not yet standardized APIs

WebSMS Only available to certified apps on B2G

GETTING HELP FROM THE COMMUNITY

Mailing lists, IRC channels, ...

- Consult the dedicated Mozilla forum :
  - as a mailing list
  - as a newsgroup
  - as a Google Group
  - as a Web feed
- #webapi on irc.mozilla.org
- WebAPI project page on wikimo

Don't forget abo

RELATED TOPICS

- The Document Object Model (DOM) is the representation of

WHO'S AWESOME?

You're awesome.

mozilla

Alucinad al mundo

Cread cosas nunca vistas

@mozilla_hispano

#FirefoxOSAppDays