Advanced Progressive Web Apps

INTEGRATING THE APP WITH HARDWARE AND PLATFORMS

Maximiliano Firtman
MOBILE+WEB DEVELOPER
@firt firt.mobi

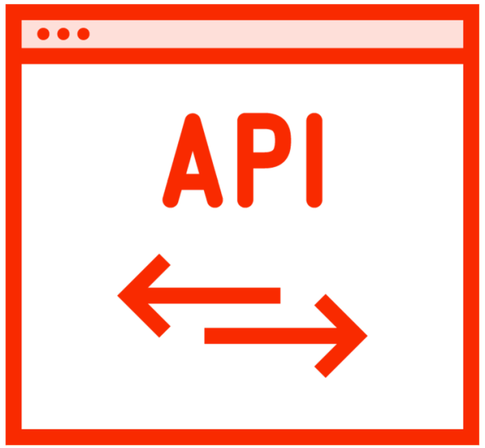
Overview

Integrating with Hardware and Platforms

- Use Hardware and Sensors with APIs
- Integrate Content with Web Share
- Create App Shortcuts
- Talk with Other Apps through URLs

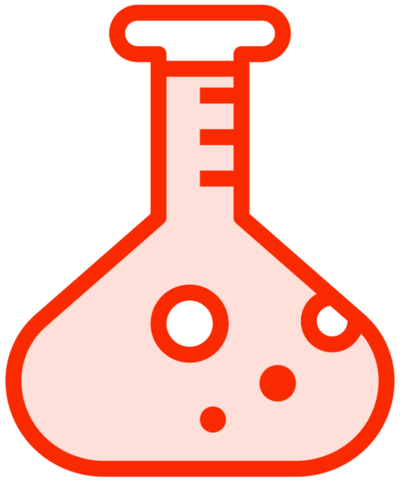
Use Hardware and Sensors with APIs

Hardware and Sensors APIs



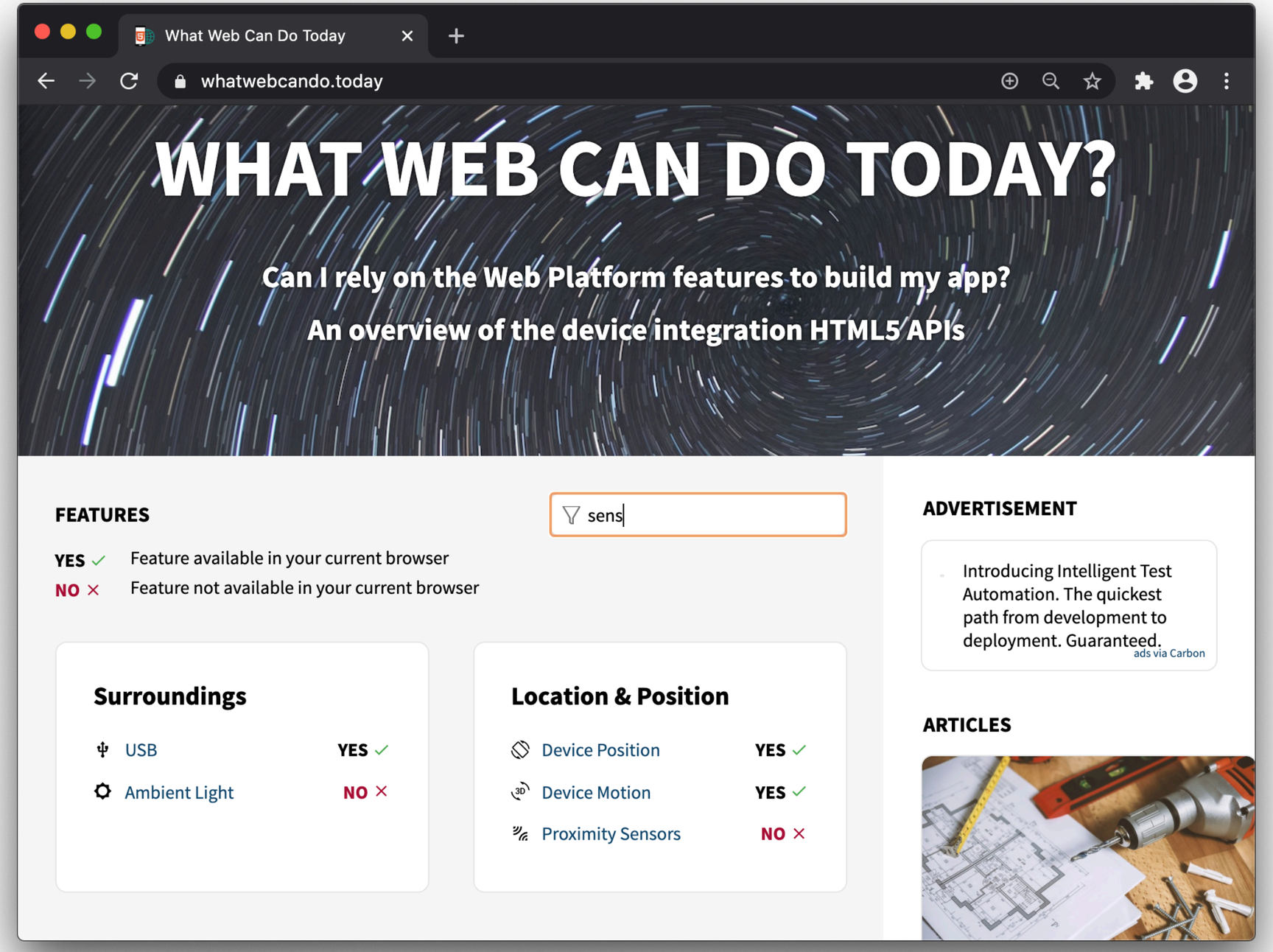
Stable and Multiplatform

Stable with Limited Support

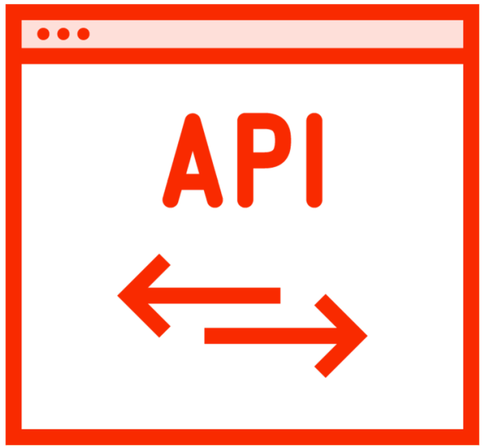


Experimental

Compatibility on different browsers change frequently


What Web Can Do Today
whatwebcando.today
WHAT WEB CAN DO TODAY?
Can I rely on the Web Platform features to build my app?
An overview of the device integration HTML5 APIs
FEATURES
sens
YES ✓ Feature available in your current browser
NO ✕ Feature not available in your current browser
Surroundings
USB YES ✓
Ambient Light NO ✕
Location & Position
Device Position YES ✓
Device Motion YES ✓
Proximity Sensors NO ✕
ADVERTISEMENT
Introducing Intelligent Test Automation. The quickest path from development to deployment. Guaranteed.
ads via Carbon
ARTICLES

Experimental APIs



Available for all users

Enabled by Settings or Flag

Enabled by Origin Trial

Google's Fugu Capabilities Project

Exposing the capabilities of native platforms to the web, while maintaining user security, privacy, and trust.

- Identify missing capability
- Initial draft of spec
- Gather feedback
- Initiate an Origin Trial
- Based on feedback, create formal spec or iterate
- If formal spec was created, launch it in stable

Check status at web.dev/fugu-status

Stable and Multiplatform APIs

- Geolocation
- Touch and Pointer
- Payment Request
- Web Authentication
- Media Capture (getUserMedia)
- Web Audio
- Speech Synthesis
- Clipboard
- OTP access
- Game Pad
- FullScreen & Picture in Picture
- Device Orientation and Motion
- Web Share

Stable with Limited Support 1/2

- WebBluetooth (GAIA)
- WebUSB
- Ambient Light
- User Idle
- Advanced Camera Controls
- Media Recorder
- Shape Detection (Bar codes, Text)
- Vibration
- Battery Status
- Device Memory

Stable with Limited Support 2/2

- Native FileSystem
- Contact Picker
- Speech Recognition
- WebXR (VR and AR)
- Screen Orientation Lock
- Wake Lock
- Presentation
- Run on Startup
- Multi Screen

Stable on one platform
doesn't mean there is a
formal standard defined yet

Experimental

- Shape Recognitions (Face)
- Web Serial
- Web NFC
- Content Indexing
- WebHID
- Media Picker
- Tabbed App Support
- WebCodecs
- Virtual Keyboard
- WebGPU
- Web MIDI

If we are distributing on App Stores we may be able to bridge to native APIs

Remember that P in PWA stands for Progressive Enhancement

Integrate Content with Web Share

Web Share

Let our PWA interact with the Share mechanism within the OS

Safari on iOS and iPadOS

Chromium-based browsers on Android, and compatible desktop OSs with share

You can Share from your PWA to other apps:

- Text
- URL
- Files

Target: Copy to Clipboard, other Apps, AirDrop, Save to File, etc.

Web Share

Sharing text to other Apps

script.js

```
navigator.share({
    title: 'Text to share',
});
```

Web Share

Sharing text to other Apps

script.js

```
if ('share' in navigator) {
    navigator.share({
        title: 'Text to share',
    });
}
```

Web Share

Checking the result of the Share operation

script.js

```
if ('share' in navigator) {
    navigator.share({
        title: 'Text to share',
    }).then( event => track('share', 'shared') )
      .catch( event => track('share', 'not shared') );
}
```

Web Share

Sharing text and links to other Apps

script.js

```
if ('share' in navigator) {
    navigator.share({
        title: 'Title to share',
        text: 'Text to share',
        url: 'https://pluralsight.com'
    });
}
```

Web Share

Sharing text and links to other Apps

script.js

```
if ('share' in navigator) {
    navigator.share({
        title: 'Title to share',
        text: 'Text to share',
        url: window.location.href
    });
}
```

Web Share

Sharing files to other Apps

script.js

```
if ('canShare' in navigator &&
    navigator.canShare({ files: [fileObject] }) ) {

    navigator.share({
        title: 'Title to share',
        files: [fileObject]
    });

}
```

Web Share Target

Chromium-based browsers on Android

Your PWA can receive data shared from other apps:

- Text
- URLs

Our PWA receives a navigation to a URL with shared data in the QueryString arguments

The spec allows also to receive POST requests that is useful for receiving:

- Complex data that changes target App
- Files

Web Share Target

Receive basic data (text and links)

app.webmanifest

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

New key in Manifest

- URL receiving data
- GET for simple data
- Map data to querystring

Web Share Target

Receive complex data that can change target's app

app.webmanifest

```
{
 "share_target": {
    "action": "/receive-action",
    "method": "POST",
    "enctype": "multipart/form-data",
    "params": {
       "text": "text",
       "url": "url"
    }
  }
}
```

Same key in manifest

- URL receiving data
- POST for complex data
- Encoding type
- Map data to Form data

Web Share Target

Receive files

app.webmanifest

```
{
 "share_target": {
    "action": "/receive-files",
    "method": "POST",
    "enctype": "multipart/form-data",
    "params": {
        "files": [
          {"name": "PNG images",
          "accept": ["image/png", ".png"]}
        ]
    }
  }
}
```

Same key in manifest
—We can process in SW
—POST for complex data
—Encoding type
—We can define an array of file types that we accept.
Per type we can define an array of extensions and MIME types

Only one *share_target* declaration is accepted in the App Manifest

The host OS will use App's name and icons from the manifest for the share dialog

Create App Shortcuts

App Shortcuts

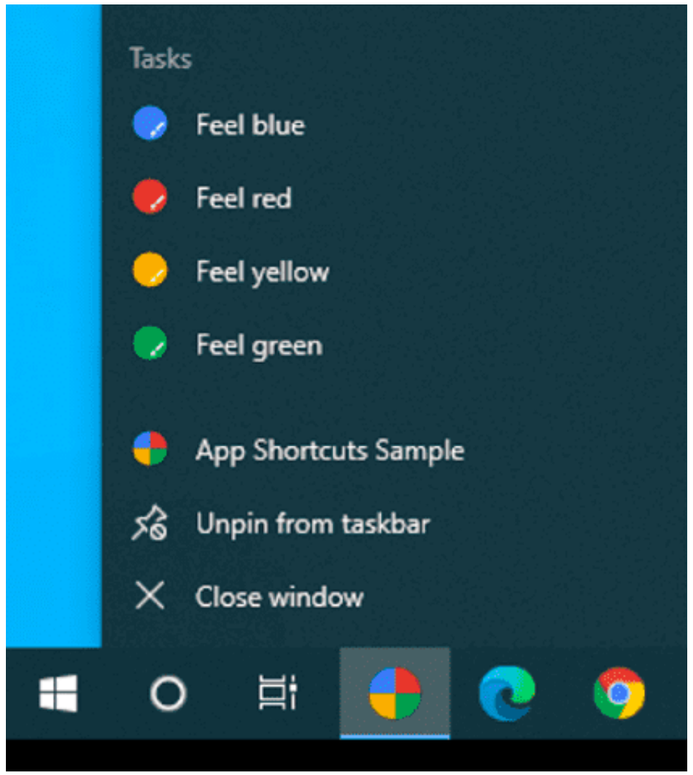
Menu items that will appear on icon launcher's contextual menu (right click or long press)

Each menu item will open a navigation within the PWA to a different URL

Menu items are defined statically per app

Changing them requires changing the App Manifest

App Shortcuts in Action



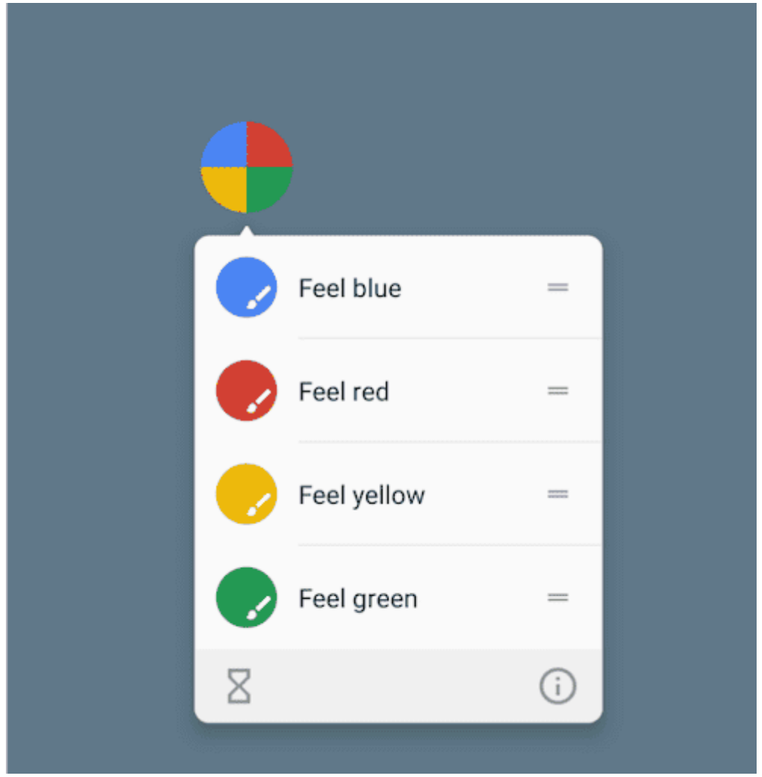


Image from web.dev

App Shortcuts

Chromium on Android and Windows

app.webmanifest

```
{
 "start_url": "/",
 "shortcuts": [{
    "name": "Open a Second Action",
    "short_name": "Second",
    "description": "",
    "url": "/second-action",
    "icons": [{
      "src": "/icons/second.png",
      "sizes": "192x192",
      "type": "image/png"
    }]
  }]
}
```

New key in Manifest

- Name
- Short Name
- Description
- URL
- Array of Icons

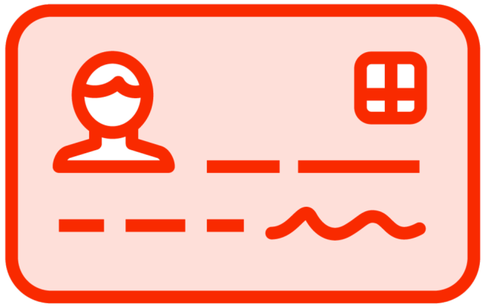
(Same rules as top-level properties)

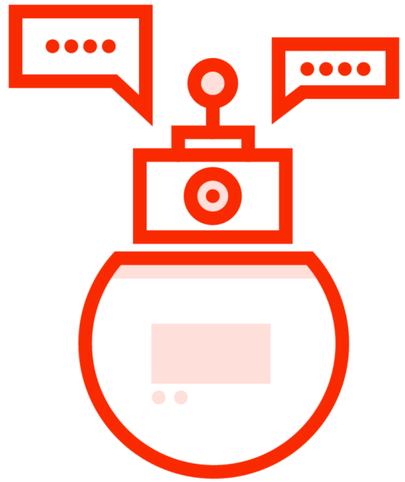
Talk with Other Apps through URLs

On a modern OS, an installed app can capture links coming from other apps before they are rendered in a browser

Link Capturing

Let's us communicate between apps in the OS





Universal Link

Normal link
https://firt.dev

Custom URI

Custom Protocol
skype://microsoft

Android Intent

Action and
Arguments

Linking from the HTML

index.html

```
<a href="url">Send data to App</a>
```

Linking from the HTML

index.html

```
<form action="url">
    <button>Open other App</button>
</form>
```

Linking from JavaScript

script.js

```
location.href = 'url';
```

On most situations, data flows one way only

Universal Links

Installed Apps that are capturing their own websites' URLs

If the app is not installed, the website will appear

The URL can be a deep link

- **Google**: Maps, YouTube, Drive, Play Store
- **Apple**: Maps, News, AppStore, iTunes, Music
- **Microsoft**: Office, Store
- **Facebook**: Instagram, Facebook, Messenger, WhatsApp
- **Other**: Twitter, Pinterest, TikTok, etc.

Custom URI scheme

Installed Apps that are capturing custom protocols

If the app is not installed, an error will arise

There is no API to check if it will work

On Android, many apps can listen to the same protocol

We pass arguments through the query string

```
protocol:action?arg1=val1&arg2=val2
```

Multiplatform URI schemes

They work on iOS, Android, iPadOS and desktop
if apps are installed

```
tel:9999999

sms:9999?body=text

mailto:email@address.com?
       subject=text&body=text

whatsapp:999999?text=text

skype:user

skype:user?call?&video=true

brave://open-url?url=url

firefox://open-url?url=url
```

- Make a phone call (use international format), +1, +34, +54
- Send a text message
- Open the default email app
  body can contain HTML on some clients
- Send a WhatsApp message with a text
  both number and text are optional
- Open the user's profile on Skype to call
- Videocall the user through Skype
- Open Brave browser in a URL
- Open Firefox browser in a URL

You can search the app you want to interact with plus "uri scheme" to see if there is any documentation

iOS and iPad OS specific URI schemes

`googlechromes://domain.com/path`	◀ Open Google Chrome for iOS on an HTTPS website
`applenewss://domain.com/path`	◀ Open Apple News on an HTTPS URL
`itms-apps://itunes.apple.com/app/id`	◀ Open the App in AppStore
`facetime-audio://phone-or-email`	◀ Audio Facetime to a user
`facetime://phone-or-email`	◀ Video Facetime to a user
`maps://url`	◀ Apple Maps (accepts same args than Google Maps)
`prefs:root`	◀ Settings
`prefs:root=SAFARI`	◀ Settings - Safari
`shortcuts://run-shortcut?name=name`	◀ Run a Shortcut with a name

Remember to use specific Apple-related URI schemes only when you are on that platform

Android Intents

Only on Android browsers

It lets us map a native Android Intent call

We can open specific apps by id or make generic calls

```
intent:
   HOST/URI-path // Optional host
   #Intent;
      package=[string];
      action=[string];
      category=[string];
      component=[string];
      scheme=[string];
      type=[string]
      S.browser_fallback_url=[string];
   end;
```

Opening Pluralsight Native app

```
<a href="intent:#Intent;package=com.pluralsight;end">
  Open Pluralsight
</a>
```

Opening Pluralsight Native app with a fallback

```
<a href="intent:#Intent;package=com.pluralsight;
         S.browser_fallback_url=https://pluralsight.com;end">
  Open Pluralsight
</a>
```

Remember to execute the intent: protocol only on Android and to check Android's native Intent documentation

Link Capturing for PWAs

Can our PWA register a URL so other apps can open us?

Capturing links from a PWA

No way to register a custom protocol

With WebAPK on Android any URL within our manifest's scope will be handled by our PWA as a Universal Link

No option to open a PWA on iOS or iPadOS from a link

On desktop, some experiments are available

Summary

Integrating with Hardware and Platforms

- Use Hardware and Sensors with APIs
- Integrate Content with Web Share
- Create App Shortcuts
- Talk with Other Apps through URLs

Summary

Advanced Progressive Web Apps

- Installing
- Using
- Updating
- Integrating

@firt firt.dev