

APIs navigateurs modernes



INITIAL



CURRENT

Sommaire

- Introduction
- Setup
- Multimédia
 - Vidéo
 - Image
- Devices
 - Géolocalisation
 - Batterie
 - Network
 - Mémoire
 - Idle
 - Senseurs

Sommaire

- PWA
 - Définition
 - Manifest
 - Service Workers
 - Web workers
 - Installation
 - Fetch
 - Cache
 - Notifications
 - Push
 - Background Sync

Sommaire

- View Transition API
 - Définition
 - Utilisation
 - Use-cases
- Wrap up and utils

Introduction

Dans un premier temps pensé pour afficher des documents statiques, le web a évolué vers des utilisations applicatives. En parallèle, l'essor du mobile au travers du développement des smartphones a posé plusieurs défis au web :

- Le besoin d'une interface s'adaptant aux différentes tailles d'écrans, densité de pixels...
- Des patterns d'utilisation mobiles et tactiles à inventer.
- Une connexion parfois intermittente ou inexistante.
- Le développement d'applications natives installables, distribuées via des Stores propriétaires. Ces applications sont souvent caractérisées par une fluidité et une intégration à l'UI native plus grande, ainsi que par l'accès à des APIs du device impossible depuis le web.

Introduction

Cependant, le web garde des atouts par rapport aux applications natives :

- Les URLs, qui permettent de lier et partager différentes pages. Une vraie solution cross-platform.
- Une communauté de développement étendue.
- Pas de dépendance à un distributeur (App Store, Play Store...)

Le web a évolué pour répondre aux enjeux de l'expérience utilisateur et de la performance. Les APIs navigateurs modernes permettent de développer des applications web de plus en plus proches des applications natives.

API

Application Programming Interface

- API d'une fonction
- API d'un composant
- API d'un navigateur
- API HTTP (REST, GraphQL, ...)

Une API est une **interface** entre deux logiciels, qui permet de communiquer entre eux. C'est donc un **contrat** entre les deux logiciels.

Une API Navigateur désigne une suite de fonctions permettant d'interagir avec le navigateur.

API Javascript

`window` : l'objet global

```
x = 6
```

```
window.x // 6
```

```
window.alert('Hello')
```

```
alert('Hello')
```

Penchons-nous plus en détails sur deux objets de `window` :

- `window.document`
- `window.navigator`

API Javascript

`document` : l'objet représentant la page

C'est la porte d'entrée du DOM.

```
document.body.style.backgroundColor = 'red'
```

- `document.querySelector`
- `document.getElementById`
- `document.createElement`
- `document.cookie`

API Javascript

``navigator`` : l'objet représentant le navigateur

- ``navigator.userAgent``
- ``navigator.geolocation``
- ``navigator.permissions``

API HTML

Le HTML est un langage balisé fortement couplé au navigateur. En ce sens, il implémente des APIs.

```

```

Le cycle de vie des APIs navigateurs

Fonctionnement attendu

- **1 - Standardisation** : les API sont définies par des organisations comme le W3C, WHATWG, ECMA, IETF, etc.
- **2 - Implémentation** : les navigateurs implémentent les API.
- **3 - Dépréciation** : les API sont dépréciées.



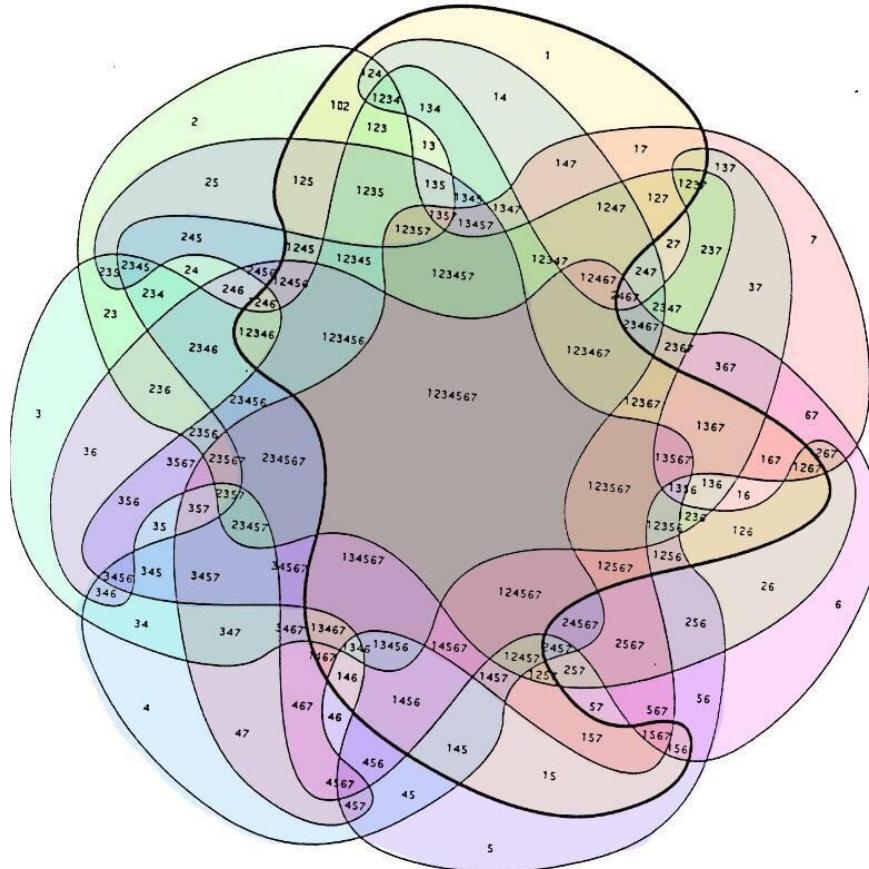
Le cycle de vie des APIs navigateurs

Réalité

- Testez la présence d'une API.
 - Pensez Progressive Enhancement / Graceful Degradation.
 - Si nécessaire, utilisez des polyfills.

<https://caniuse.com/> est votre ami.

<https://whatwebcando.today/>



Setup

Exercices et démos

<https://github.com/ecvdbdx/browsers-apis>

Devtools

- Ouvrir les devtools: cmd/ctrl + option + i
- Control panel: cmd/ctrl + shift + p



Multimédia

Vidéo

```
`npm run dev:video:exo`
```

PARTIE 1 :

- Implémenter un custom player (lecture, pause, volume, mute, progression)

PARTIE 2 :

- Mettre en pause la vidéo quand elle quitte le viewport
- Relancer la vidéo quand elle rentre dans le viewport

BONUS

- Respecter `prefers-reduced-motion`

Multimédia

Vidéo

- [MDN - Video and audio APIs](#)
- [MDN - intersectionObserver](#)
- [Smashing Mag - Respecting user motion preference](#)

Multimedia

Images

```
`npm run dev:images:exo`
```

- Implémenter un lazy loading d'images
 - Utilisez la balise `picture`
 - Utilisez les attributs `loading` et `fetchPriority`

Multimedia

Images

- Web.dev - Browser level images lazy loading
- Demo
- Smashing Mag - Fetch Priority

Pouvez-vous penser à une implémentation de lazy loading en Javascript ?

Geolocation

```
`npm run dev:geolocation-v1:exo`
```

- Implémenter un système de géolocalisation permettant de savoir si l'utilisateur se trouve à l'ECV Digital

Geolocation

- [MDN - Geolocation API](#)

Device

```
`npm run dev:device:exo`
```

- Remplir chaque valeur du tableau pour en savoir plus sur notre utilisateur

Device

Secure Context

Certaines APIs nécessitent un contexte sécurisé (HTTPS).

- [MDN - Secure Contexts](#)

Une pincée de soupçons

Les valeurs reçues sont simplement indicatives et sont dépendantes du device.

Device

Use-cases

Les APIs de batterie, network et mémoire sont notamment utiles pour:

- Limiter les appels réseaux en fonction du network ou de la batterie
- Limiter les animations (Javascript, WebGL, CSS intensives...) en fonction de la mémoire disponible et de la batterie
- Proposer un mode "low power" à l'utilisateur

Idle detection

- [MDN - Idle Detection API](#)

Use-cases

- Limiter les appels réseaux en fonction de l'activité de l'utilisateur.
- Faire le lien avec l'application native pour n'activer les notifications que lorsque l'application desktop est "idle". (Ex: Slack)

Senseurs

- Chrome Developers - Sensors for the web



PWA

- Responsive (adaptation aux tailles d'écrans, densité de pixels, etc...) Indépendante du réseau (capacité offline)
- App-like (transitions, fluidité...)
- Capacité à se maintenir à jour sans action de l'utilisateur
- Sécurisée (HTTPS)
- Découvrable (via le manifest lors de la navigation dans un navigateur) Engageante (capacité à pousser de l'information à l'utilisateur via notifications) Installable
- Partageable via URL sans installation

PWA

Caractéristiques techniques

- HTTPS : A l'exception de localhost pour le développement, les PWAs doivent être servies via HTTPS pour que le service worker puisse être installé
- Service worker : Le service worker est un proxy côté client entre le site et le réseau
- Web app manifest : Le manifest permet d'installer l'application sur le bureau, et donne au navigateur des règles d'affichage et de comportements.

PWA

Web Technologies VS Web Platform

Beaucoup de solutions existent pour développer des applications installables avec des technologies web :

- Encapsulation (Cordova...)
- Applications hybrides (React Native, Ionic, Electron...)

Ces solutions permettent de développer une application en utilisant les technologies web (HTML / CSS / JS), mais elles ne peuvent pas exister sur le web sans ajustements.

PWA

Service Worker

Le Service Worker est un worker Javascript (web worker) qui s'installe sur le navigateur à l'ouverture du site. Il agit comme un proxy côté client, qui peut intercepter les requêtes selon des règles définies et décider comment y répondre :

- Mise en cache / service depuis le cache
- Réponse custom
- Mise en attente de requêtes qu'il peut rejouer lorsque le serveur est atteignable
- Il peut se mettre à jour dans toutes les fenêtres qu'il contrôle

PWA

Web App Manifest

Le Manifest est un fichier JSON contenant des clés et valeurs définies par l'API du Manifest, qui permettent au navigateur d'identifier une application installable. Il est inséré dans un lien de la `<head>` de la page et contient entre autres :

- Set d'icônes à afficher sur le bureau
- Nom
- Règles d'affichage : défaut, standalone...
- Orientation, couleurs...

PWA

Installation

```
`npm run dev:pwa:00`
```

Pour permettre l'installation de la PWA sur le bureau, plusieurs pré-requis :

- Servir l'application en HTTPS (ou localhost)
- Déclarer les champs `name`, `short_name`, `icons` et `start_url`
- Configurer un Service Worker pour un fonctionnement offline

PWA

Cache API

L'API Cache permet de stocker des objets REQUEST / RESPONSE dans le cache du navigateur. Elle est accessible aussi bien depuis le Service Worker que depuis le contexte de l'application.

Cache API

PWA

Cycle de vie du Service Worker

```
`npm run dev:pwa:01`
```

- Téléchargement
- Installation
- Wait
- Activation

PWA

Update des fichiers en cache

```
`npm run dev:pwa:02`
```

UTILISATION D'UNE STRATÉGIE NETWORK-FIRST

Privilégez une stratégie network-first pour toujours servir les derniers fichiers.

VERSIONING DU CACHE

Utilisez un versioning du cache pour forcer le rechargeement des fichiers.

UTILISATION D'UN PROCESSUS DE BUILD

Il est recommandé d'utiliser un processus de build pour générer un fichier de cache.

PWA

Inspecter et updatrer le Manifest

<chrome://internals/web-app>

Les changements au Manifest devraient être reflétés en un ou deux jours. Pour vérifier en développement les changements :

- Quitter la PWA
- <chrome://restart>
- Relancer la PWA (installation du nouveau Manifest)
- Quitter puis ouvrir à nouveau

PWA

Communication entre le Service Worker et la page

```
`npm run dev:pwa:03`
```

La communication entre le document et le Service Worker peut se dérouler via l'API `postMessage`, implémentée des 2 côtés.

[MDN - postMessage](#)

PWA

Offline page

```
`npm run dev:pwa:04`
```

A minima, un Service Worker peut être utilisé pour servir une page d'erreur lorsque le réseau n'est pas disponible.

PWA

Caching dynamique

```
`npm run dev:pwa:05`
```

On ne peut pas toujours lister l'intégralité des pages à mettre en cache (comme lorsqu'on utilise un CMS).
Dans ce cas, la mise en cache peut s'effectuer dynamiquement, au moment de la visite de la page.

PWA

Ressources

LIVRES

- Going offline - <https://abookapart.com/products/going-offline>
- Progressive Web Apps - <https://abookapart.com/products/progressive-web-apps>
- Resilient web design - <https://resilientwebdesign.com/>

SITES

- <https://serviceworke.rs/>
- <https://web.dev/progressive-web-apps/>

Notifications

Suite à des abus, les notifications doivent être déclenchées par une action de l'utilisateur (click, touch, etc...), et le navigateur doit récolter sa permission.

- [MDN - Notifications API](#)

Online / Offline

- MDN - onLine