PWAs with Angular

Blurring the lines between native and web apps

George Kalpakas
@gkalpakas

George Kalpakas

- Angular core team
- gkalpak
- @gkalpakas

What

is a **Progressive Web App** (PWA)?

PWA is a **collection of characteristics**:

- Progressive
  Work for any user. Progressively enhanced.
- Responsive
  Fit any form factor.
- Linkable
  Zero-friction, zero-install, easy to share.
  The social power of URLs matters.
- Safe
  Served via TLS to prevent snooping.
- App-like interactions
  Shell + Content app model to create “appy” navigations & interactions.
- Fresh
  Transparently always up-to-date.
- Discoverable
  Identifiable as “apps”, allowing search engines & app stores to find them.
- Connectivity independent
  Work offline or on unreliable connection.
- Re-engageable
  Access the re-engagement UIs of the OS.
- Installable
  Integrate with OS (e.g. on home screen).
  “Keep” useful apps without the app store hassle.

> PWAs: Escaping Tabs Without Losing Our Soul
> Alex Russell, 2015

@gkalpakas

Why
should **you** care?

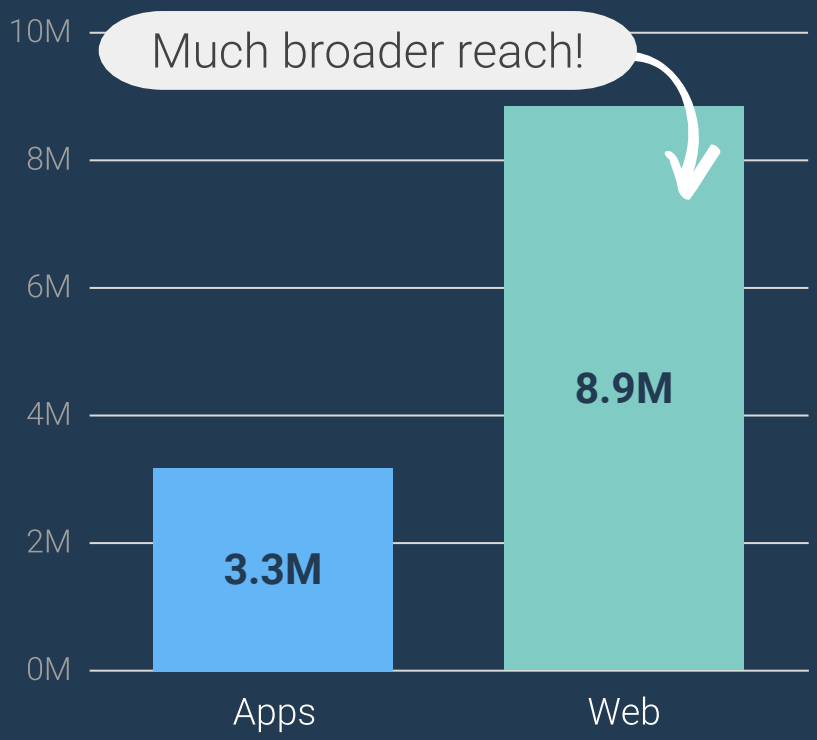
The better **User eXperience** an app offers...

...the better it can **fulfill its purpose**.

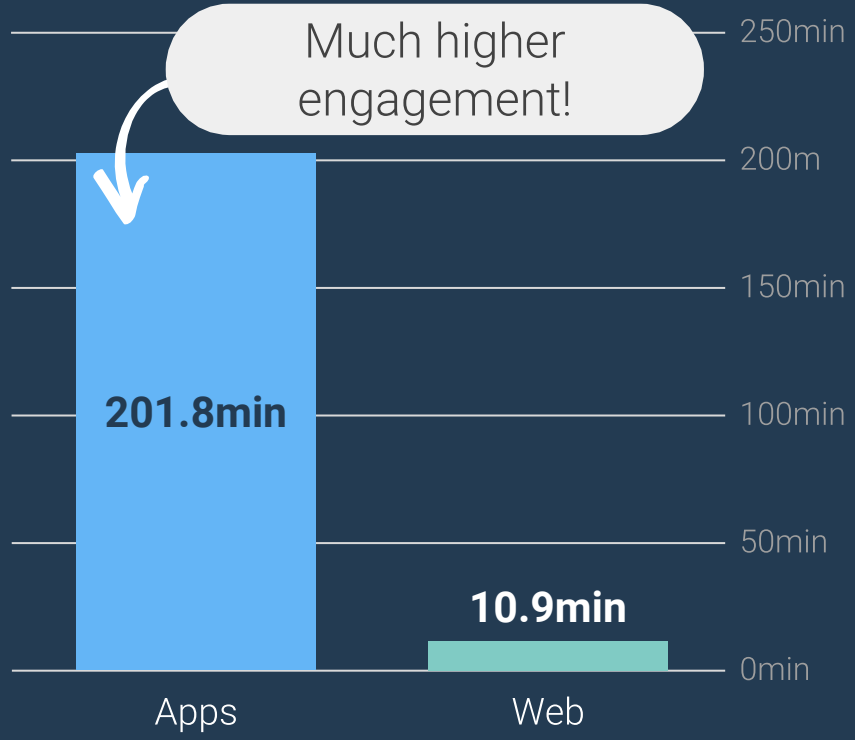
(For any app type: commercial, gaming, productivity, etc.)

Mobile web vs native apps

Source: Comscore Mobile Metrix®, 2015-2017



Monthly unique visitors



Avg. monthly minutes per visitor

@gkalpakas

✍️ Write **once**...

...run **anywhere**? 🏃

Use the Platform, Luke.

(* with caveats but still...)

@gkalpakas

In Summary

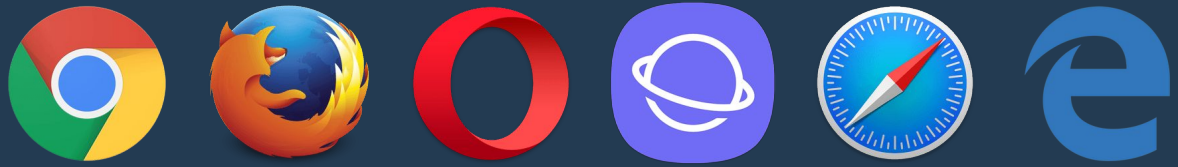
A **PWA** offers:

- Great UX.
- Broad reach.
- High engagement.
- Low development/maintenance cost.

Where

can PWAs be **used**?

Browsers:



Browsers: 

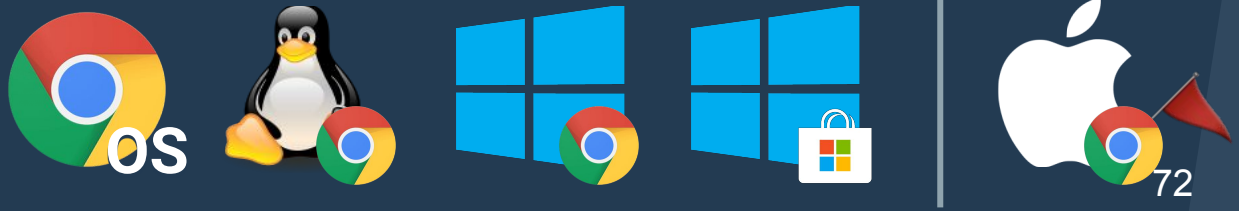
Mobile: 

Browsers | Mobile | **Desktop**

Browsers:

Mobile:

Desktop:



@gkalpakas

Browsers | Mobile | Desktop

Browsers:

Mobile:

Desktop:

Reminder

Progressive enhancement ➡ **Still work** on old browsers

How

can we **build** a PWA?

PWAs have to be:

- Progressive
- Responsive
- Linkable
- Safe
- App-like interactions

- Fresh
- Discoverable
- Connectivity independent
- Re-engageable
- Installable

Using
traditional technologies

New browser APIs:

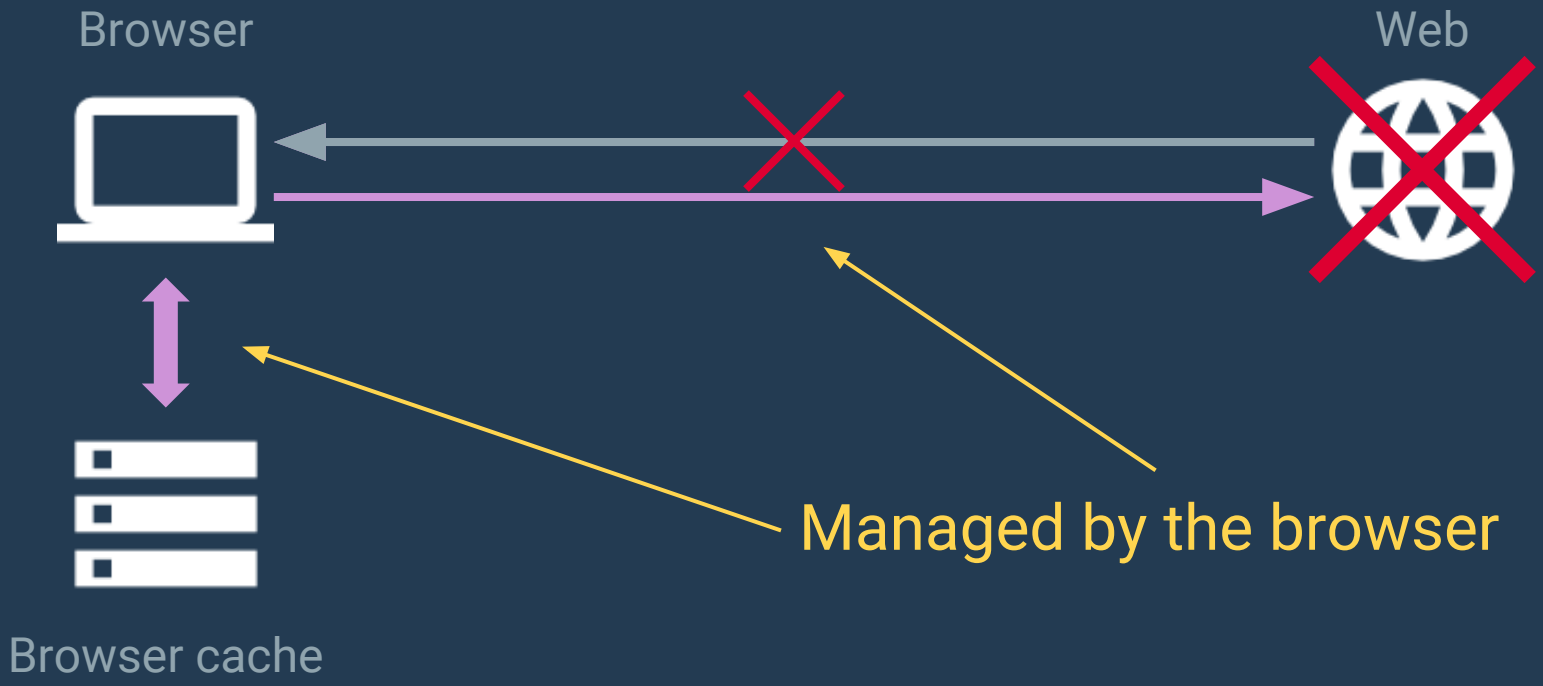
Service Worker + Fetch	✓ Fresh
Web App Manifest	✓ Discoverable
Service Worker + Cache + IndexedDB	✓ Connectivity independent
Service Worker + Clients + Notifications + Push	✓ Re-engageable
Service Worker + Web App Manifest	✓ Installable

New browser APIs:

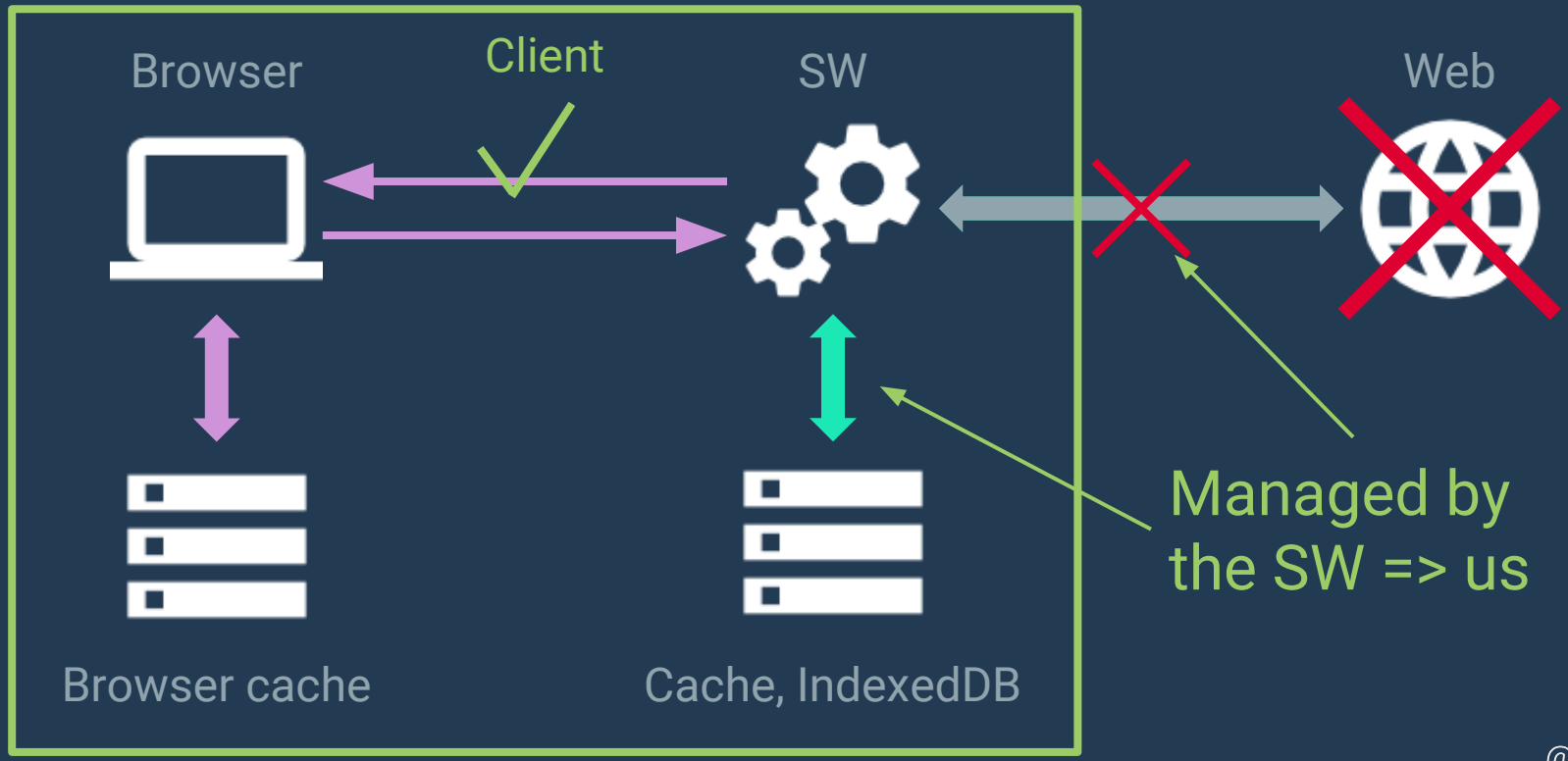
Service Worker + Fetch	✓ Fresh
Web App Manifest	✓ Discoverable
Service Worker + Cache + IndexedDB	✓ Connectivity independent
Service Worker + Clients + Notifications + Push	✓ Re-engageable
Service Worker + Web App Manifest	✓ Installable

Service Worker *(SW)*

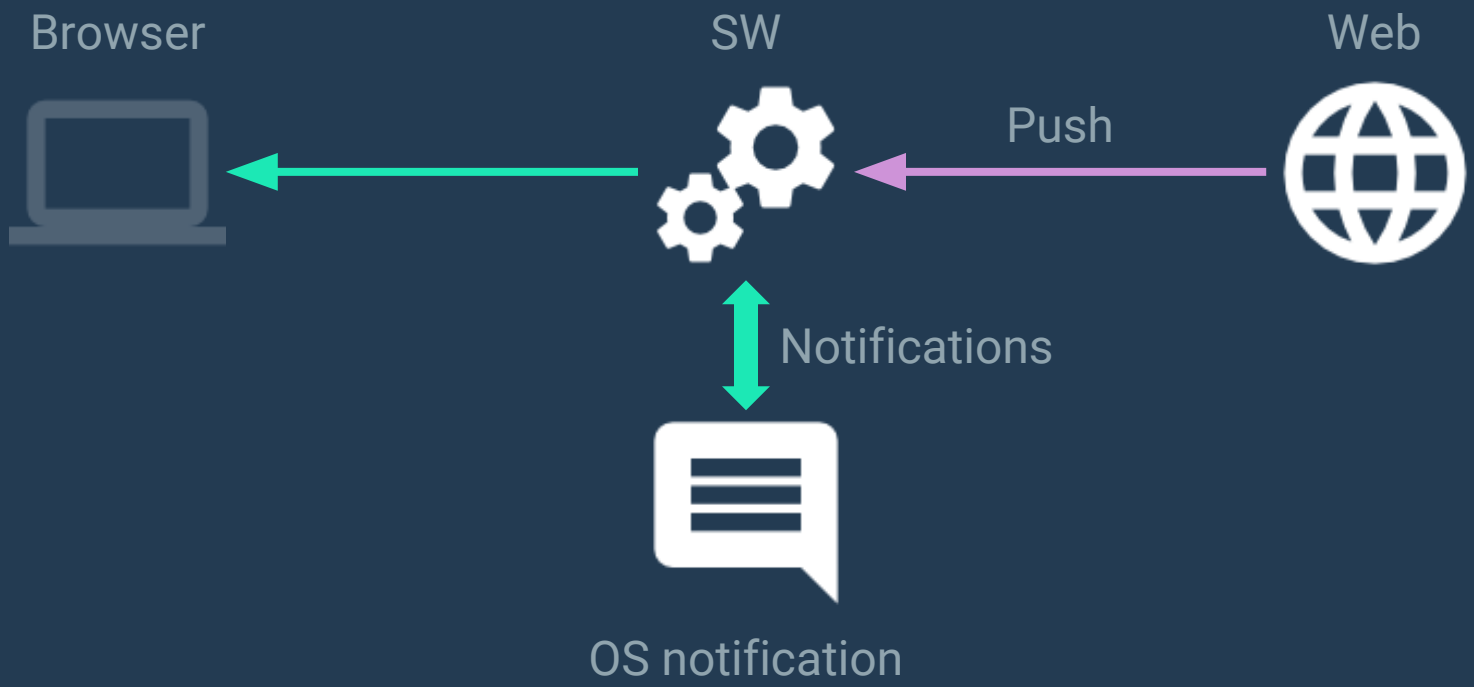
Without service worker



With service worker



Server-initiated push



SW code: Installation / Activation

```
self.addEventListener('install', event => {
  // E.g. populate caches with new resources.
  event.waitUntil(doAsyncStuff());
});

self.addEventListener('activate', event => {
  // E.g. clean up old caches.
  event.waitUntil(doAsyncStuff());
});
```

SW code: Requests

```
self.addEventListener('fetch', event => {
  // E.g. forward the request to the server
  // or retrieve a response from the cache
  // or come up with a random response ¯\_(ツ)_/¯
  event.respondWith(someResponse);
});
```

SW code: Other events

```
self.addEventListener(
  'message/notificationclick/push/...',
  event => {
    // Handle the event.
    event.waitUntil(doAsyncStuff());
  },
);
```

Web App Manifest

Web App Manifest:

- Simple JSON text file.
- Provides information about the app (e.g. name, icons, description).
- Helps identify a web app as PWA.
- Helps better integrate with OS.

manifest.json

```
{
  "name": "My PWA",
  "description": "...",
  "start_url": ".",
  "display": "standalone",
  "theme_color": "orchid",
  "icons": [...],
  ...
}
```

index.html

```
<link
  rel="manifest"
  href="manifest.json"
  />
```

PWA-related Tooling

Note of caution

Low-level, powerful APIs:

- Easy to shoot oneself in the foot.

Using tools ensures:

- Battle-tested solutions.
- Best practices.
- Not re-inventing the wheel.

Building PWAs (1/3)

PWABuilder

- Online wizard and CLI tool for generating PWA from any site.
- Additionally supports packaging PWAs as native apps (via Cordova).
- Good for initial experimentation.

Easy to get started, extensible

Limited feature-set

Building PWAs (2/3)

Workbox

- Set of libraries and Node modules for building PWAs.
- Plugin-based architecture.
- Supports extending existing SW.

Feature-rich, extensible, composable

Relatively complicated set-up

Building PWAs (3/3)

@angular/service-worker

- Angular's SW implementation.
- Offers several features out-of-the-box.
- Angular & Angular CLI integration.

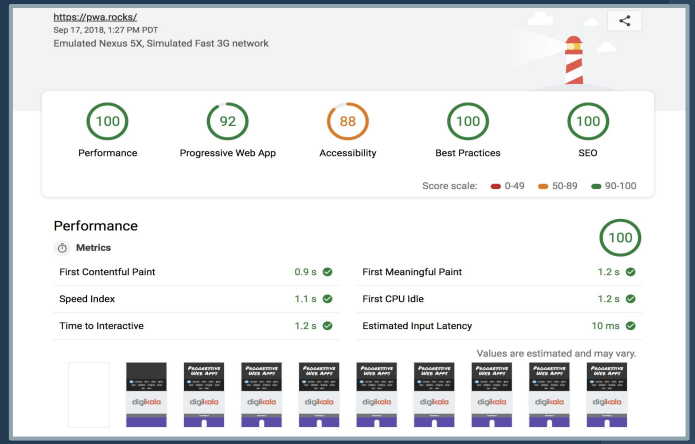
No coding required, easy to use from Angular app (DI, services)

Not extensible/composable

Testing PWAs (1/2)

Lighthouse

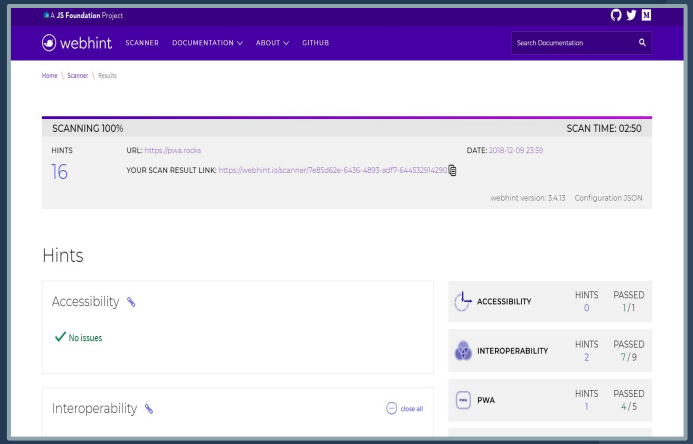
- Open-source, automated tool for improving quality of web pages.
- Audits for PWA, performance, accessibility, and more.
- Built into Chrome.
  Useful during development.
- CLI tool / Node module.
  Useful for CI.



Testing PWAs (2/2)

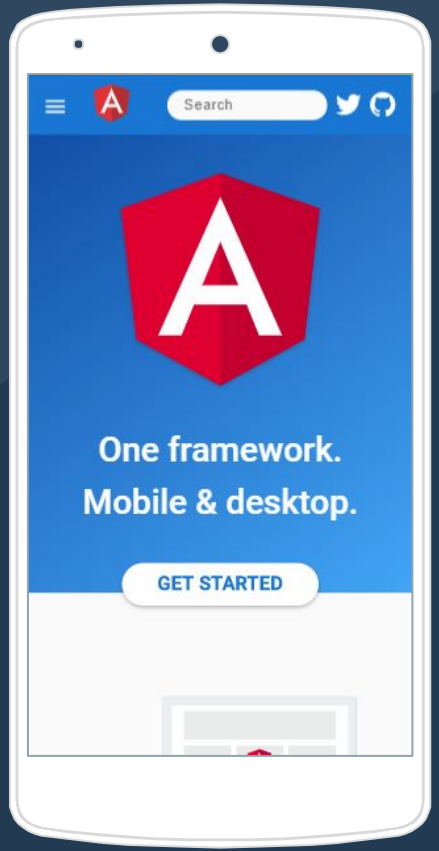
Webhint

- Open-source, community-driven linting tool.
- Helps improve speed, accessibility, security, PWA, and more.
- Checks for best practices and common errors.
- Online scanner.
  Useful during development.
- CLI tool.
  Useful for CI.



Demo time!

https://github.com/gkalpak/ng-athens-pwa-demo



Thank You!

Slides | bit.ly/gk-ngathens-pwa

George Kalpakas
@gkalpakas

Resources: Browser APIs

Stable APIs:

- Cache API
- Clients API
- Fetch API
- IndexedDB API
- Notifications API
- Push API
- ServiceWorker API
- Web App Manifest

Proposed/Draft APIs:

- Background Fetch
- Background Sync
- Writable Files

Miscellaneous:

- Is Service Worker ready?
- Project Fugu (successor of Project Fizz)

Resources: Tooling/Docs

Tooling:

- Angular SW: Intro
- Angular SW: Config
- Workbox
- PWABuilder

- Lighthouse
- Webhint (prev. Sonarwhal)

Guides/Blog posts:

- PWAs
- Desktop PWAs
- PWAs and Windows 10

- Debugging Service Workers

- PWAs: Escaping Tabs Without Losing Our Soul
  Alex Russell, 2015 - Introducing PWAs