



Felix Gessert

Service Workers:

The Technology Behind Progressive Web Apps

Frontend



code.talks

What we are going to cover.

PWAs



- Core Features
- Building Blocks
- Implementation

Service Workers



- Lifecycle
- Network Interception
- Caching Strategies

Use Case



Service Workers
in Production
at Baqend

Why do(n't) we love native apps?

Progressive Web Apps

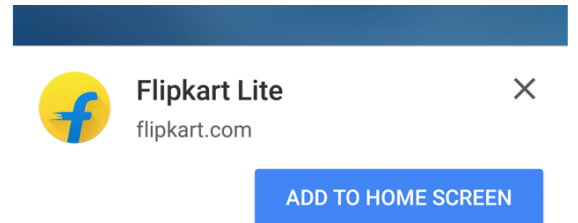
Combine the best from **native** and **web apps**.



A person wearing a white long-sleeved shirt is holding a smartphone with both hands. The person's left wrist is visible, wearing a silver-toned watch with a white face and a metal link bracelet. The watch face has 'ck' and 'Calvin Klein' printed on it. The background is a dark, textured surface. The text 'What are Progressive Web Apps?' is overlaid in the center in a large, white, sans-serif font.

What are Progressive Web Apps?

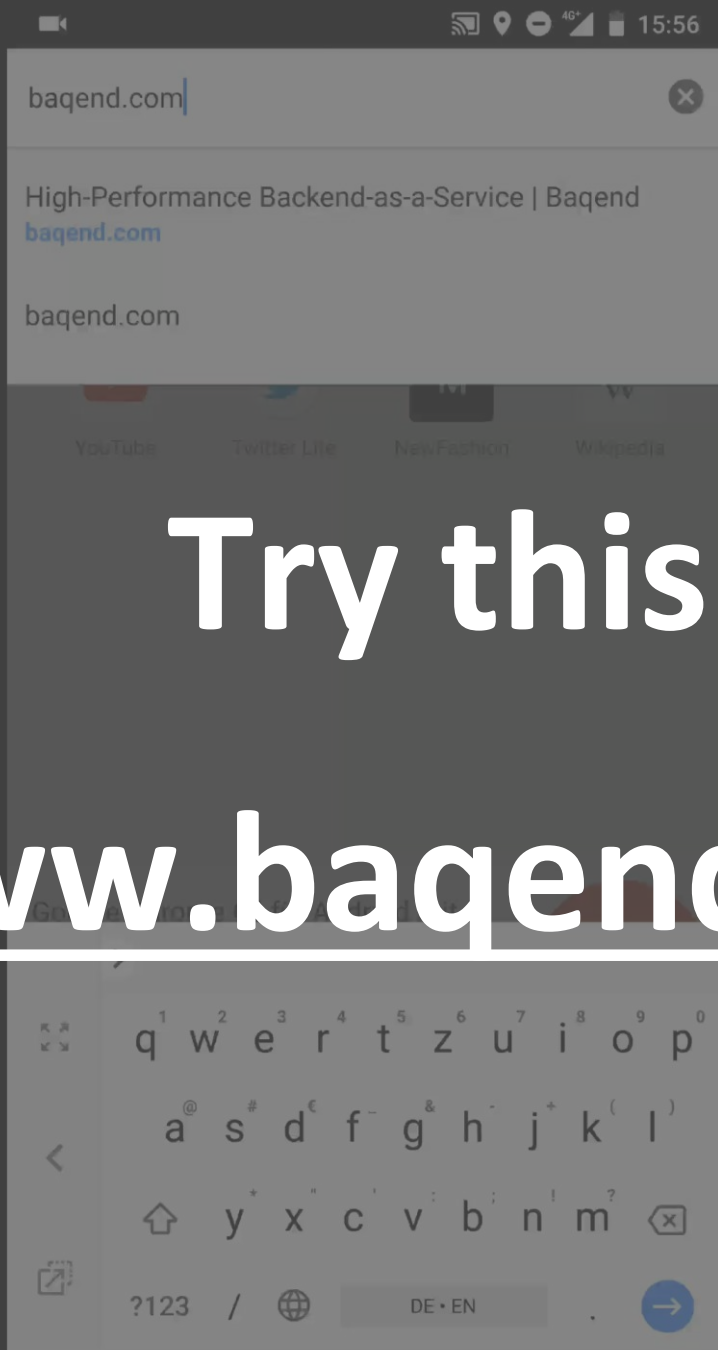
Progressive Web Apps (**PWAs**)



Fast **Loads**
through Caching

Offline Mode
(Synchronization)

Add-to-**Homescreen**
and **Push Notifations**



Try this:

www.baqend.com

Advantages of PWAs



Discoverable

E.g. in search engines



Installable

Easy access from home screen



Linkable

Link into apps through URLs



Network independant

Offline mode



Progressive

Enhance on capable browsers



Re-engageable

Engage through Web Push



Responsive

Fit any form factor



Safe

HTTPS & recognizable URLs





These apps aren't packaged and deployed through stores, they're just **websites that took all the right vitamins.**



Alex Russell, Google

Building Blocks of PWAs

PWAs are **best practices**
and **open web standards**



1. **Manifest**

Progressively enhance
when supported



2. **Service Worker**

Implementing PWAs

PWAs are **best practices**
and **open web standards**

Progressively enhance
when supported

1. **Manifest** declares Add-to-Homescreen:

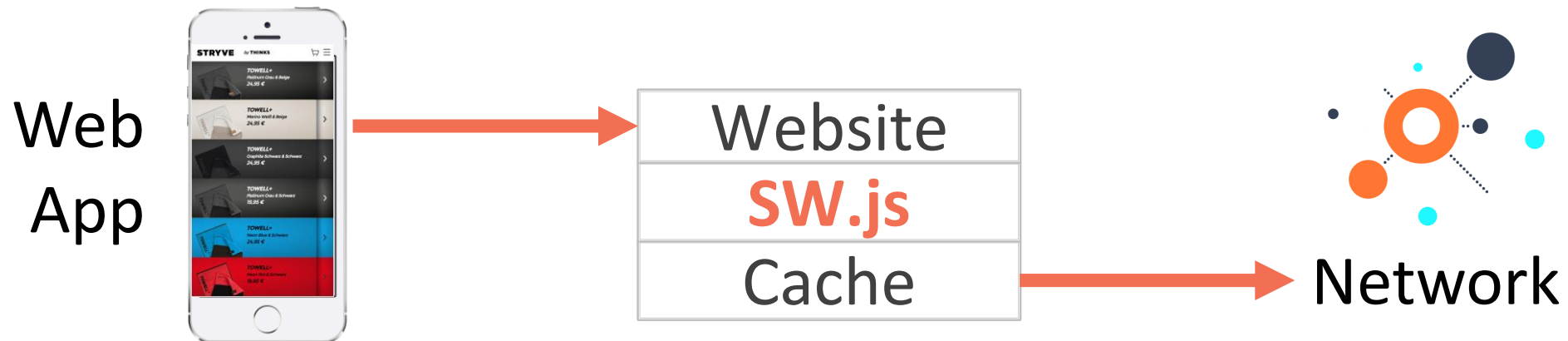
```
<link rel="manifest" href="/manifest.json">
{
  "short_name": "Codetalks PWA",
  "icons": [
    {"src": "icon-1x.png", "type": "image/png", "sizes": "48x48"}],
  "start_url": "index.html?launcher=true"
}
```


Implementing PWAs

PWAs are **best practices**
and **open web standards**

Gracefully degrade when
not supported

2. **Service Workers** for caching & offline mode:

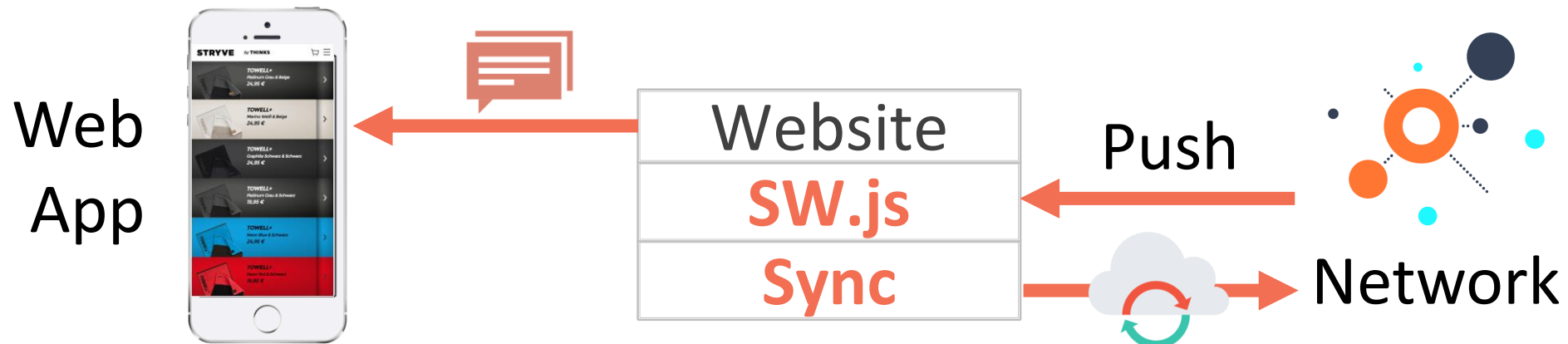


Implementing PWAs

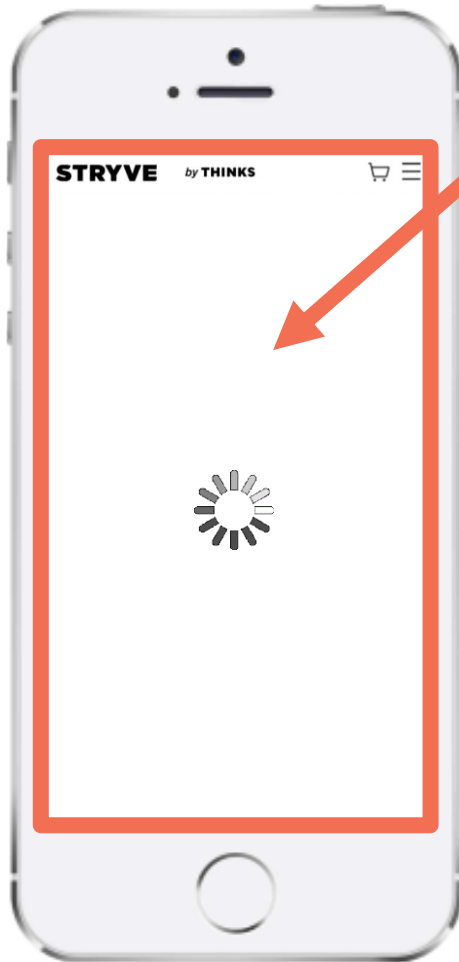
PWAs are **best practices**
and **open web standards**

Progressively enhance the
user experience

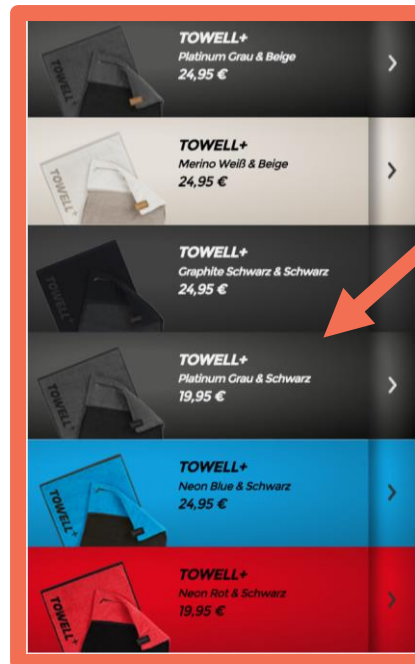
3. Add **Web Push** and **Background Sync**:



Typical Architecture: App Shell Model



App Shell: HTML, JS, CSS, images with app logic & layout



Content: Fetched on demand & may change more often

What is the future and vision of Progressive Web Apps?



Integrate **payment**.

[Samples](#)

Background

PaymentRequest Free Shipping Sample
googlechrome.github.io

[Order summary](#)

Donation USD \$55.00

[Shipping](#)

Eiji Kitamura
Ripping Hills Mori Tower 4...ato-ku, TOKYO, 106-6144
1111-1111
Free shipping worldwide

[Payment](#)

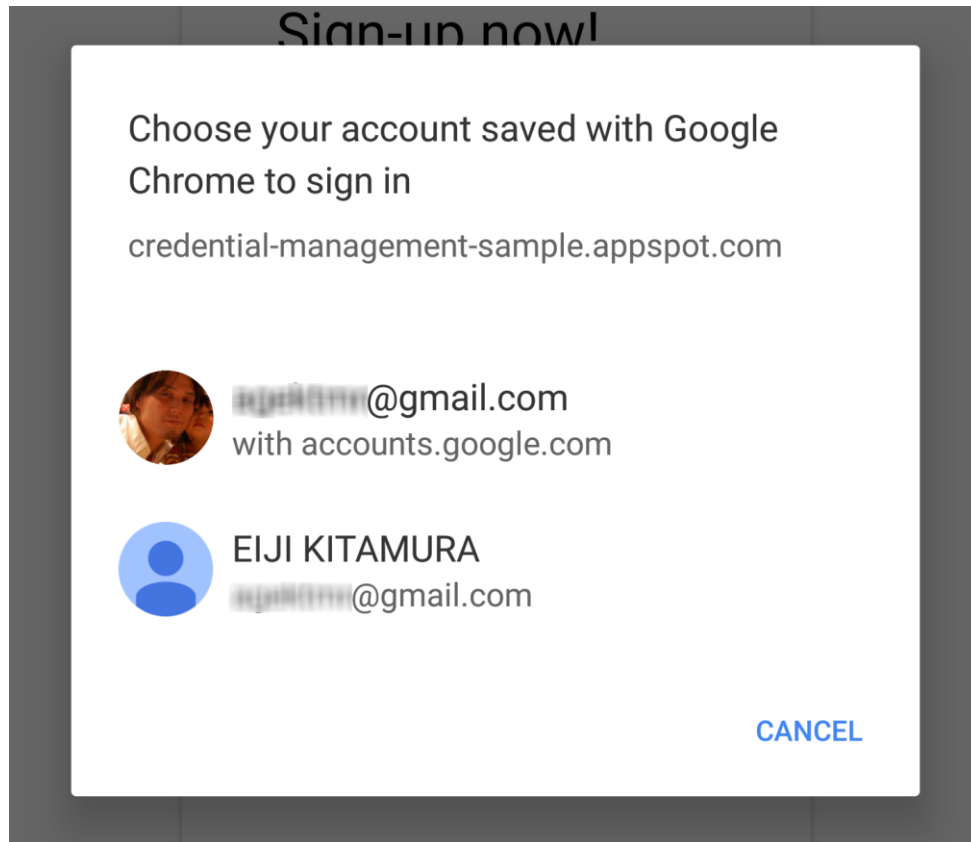
Visa ••••4242, Eiji Kitamura

[EDIT](#) [PAY](#)

Web Payment APIs

- Goal: replace traditional **checkout** forms
- Just ~10 LOC to implement **payment**
- Vendor- & Browser-**Agnostic**

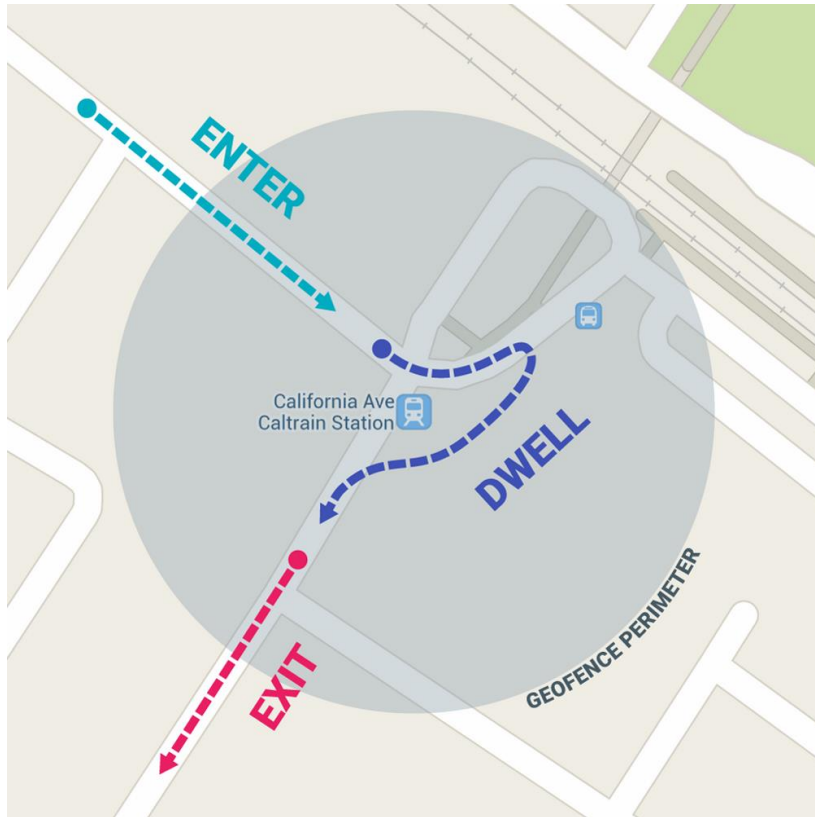
Manage **users** and **logins**.



Credentials Management API

1. Click **Sign-in** → Native Account Chooser
2. Credentials API **stores** information for future use
3. **Automatic** Sign-in afterwards

Leverage **geolocation**.



Geofencing

- **Notify** web app when user leaves or enters a defined area
- Requires **permission**

Build **conversational** interfaces.



Web Speech API

Native Speech Recognition in the Browser:

```
annyang.addCommands({  
  'Hello Code.talks': () => {  
    console.log('Hello you.');  }  
});
```

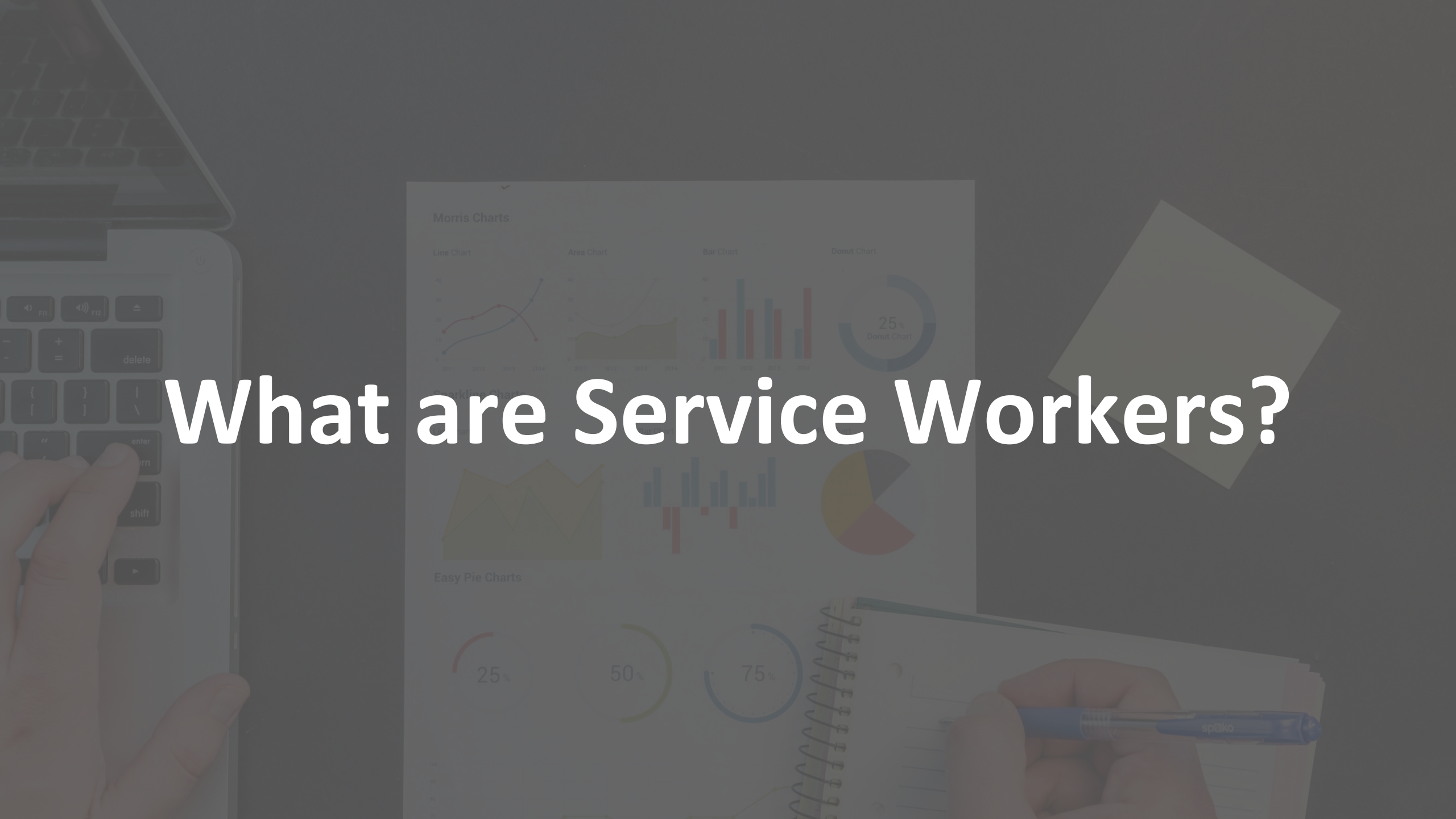

Seamless **sharing** between apps.



Web Share API

- **Share** site through native share sheet UI
- Service Worker can register as a **Share Target**

What are Service Workers?



What are **Service Workers**?



Programmable **Network Proxy**, running as a separate **Background Process**, without any **DOM Access**.

What do **Service Workers** do?



- **Cache** Data (CacheStorage)
- **Store** Data (IndexedDB)
- Receive **Push**
- Respond when **Offline**

What do **Service Workers** do?



- **Intercept** HTTP Requests
- **Sync** Data in Background
- Hide **Flaky Connectivity** from the User

Browser Support for Service Workers

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			49						
			63						
			67						
		61	68	11.1	11.2				4
11	17	62	69	12	11.4	all	69	11.8	7.2
	18	63	70	TP	12				
		64	71						
			72						

Supported by **>85%** of browsers.

Requires **TLS Encryption**.

Late, but all in: Microsoft

Publish PWAs to
Microsoft Store



**PWA
BUILDER**

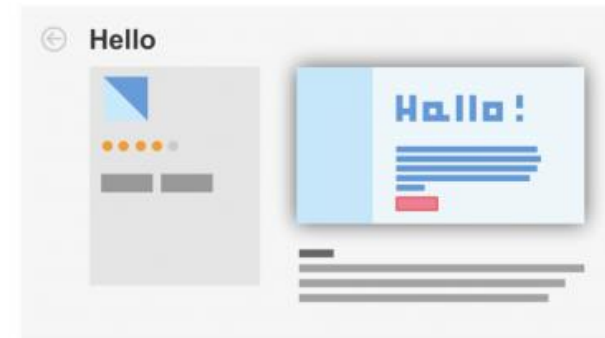
or



**Bing Crawl
PWAs**



**Convert to
AppX**



Microsoft Store



How are **Service Workers** registered?



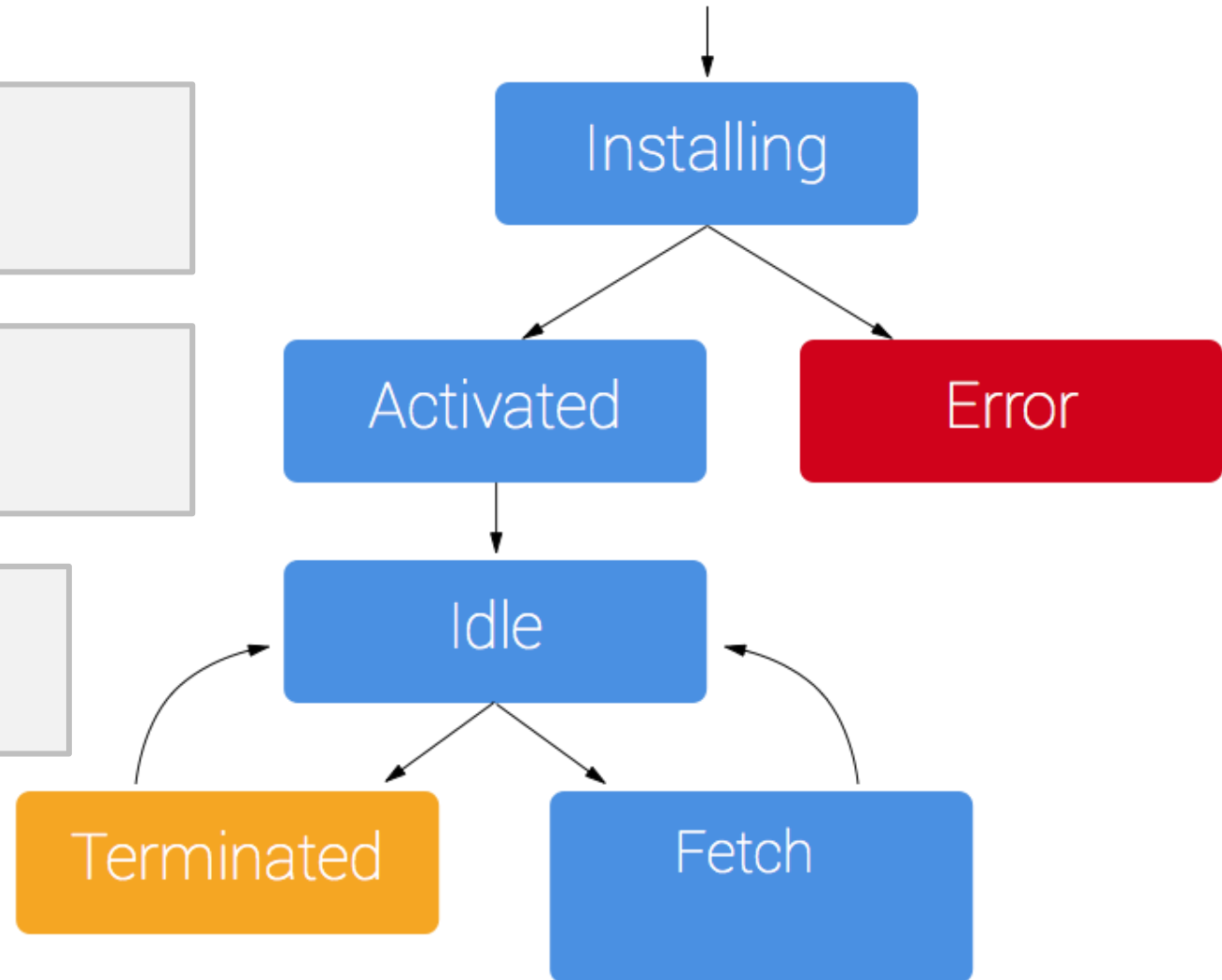
```
<script>  
  navigator.serviceWorker.register('/sw.js');  
</script>
```


What does the **lifecycle** look like?

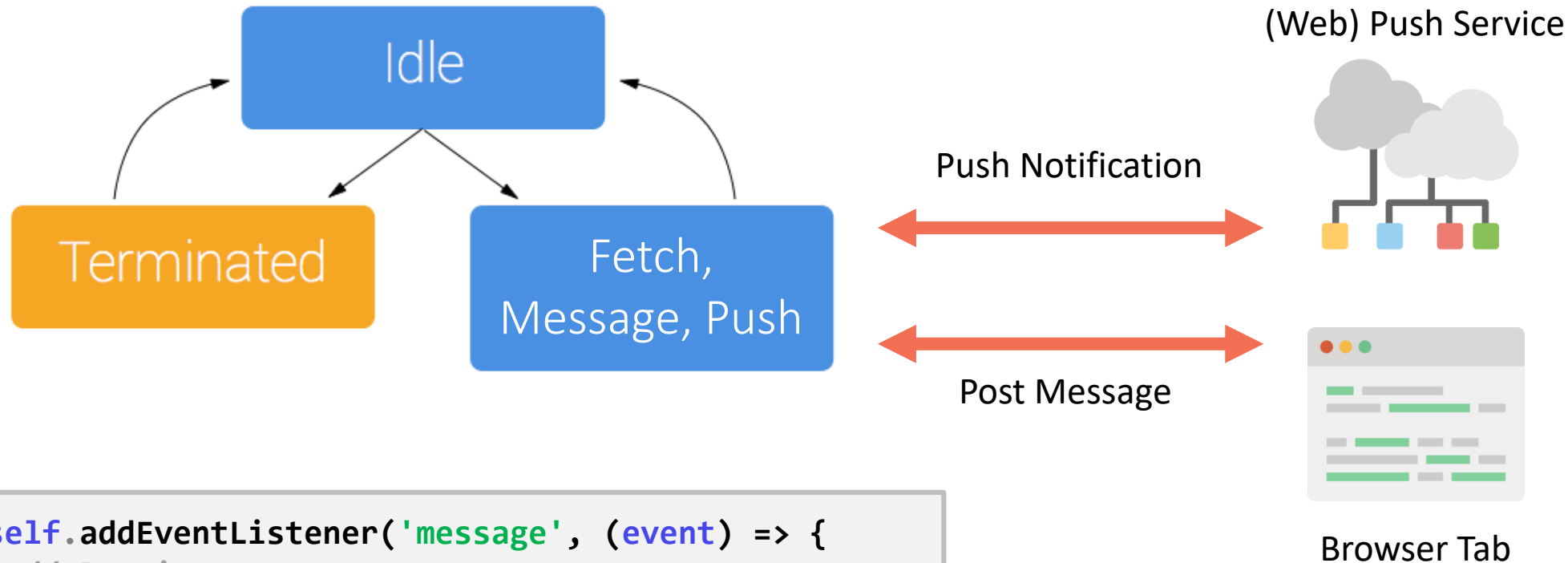
```
self.addEventListener('install', (event) => {  
  // Perform install steps  
});
```

```
self.addEventListener('activate', (event) => {  
  // Perform activate steps  
});
```

```
self.addEventListener('fetch', (event) => {  
  // React to fetch event  
});
```



How to **communicate** with Service Workers?



```
self.addEventListener('message', (event) => {  
  // Receive message  
});
```

```
// Send message to browser tab  
const client = await clients.get('id');  
client.postMessage(someJsonData);
```

```
self.addEventListener('push', (event) => {  
  // Receive push notification  
});
```


Intercepting Network Requests



```
self.addEventListener('fetch', (event) => {  
  // React to fetch event  
  const { url } = event.request;  
  event.respondWith(async () => {  
    const request = new Request(url.replace('.com', '.de'))  
    const response = await fetch(request);  
    const text = await response.text();  
    const newText = text.replace('Goethe', 'Schiller');  
    return new Response(newText, { status: 200 });  
  })();  
});
```

There is so much you can do:

- **Rewrite** Requests
- **Change** Responses
- **Concat** Responses
- **Cache** Responses
- **Serve** Cached Data
- ...

Service Worker **Scope**



Scope determines which requests go to the Service Worker

```
// Default (and maximum) scope is location of Service Worker  
// Gets all requests starting with '/path/'  
navigator.serviceWorker.register('/path/sw.js');
```


Service Worker **Scope**



Scope can be restricted but not widened

```
// Scope option can further limit which requests got to Service Worker  
// Gets all requests starting with '/path/subpath/'  
navigator.serviceWorker.register('/path/sw.js', { scope: '/path/subpath/' });
```


Service Worker **Persistence**

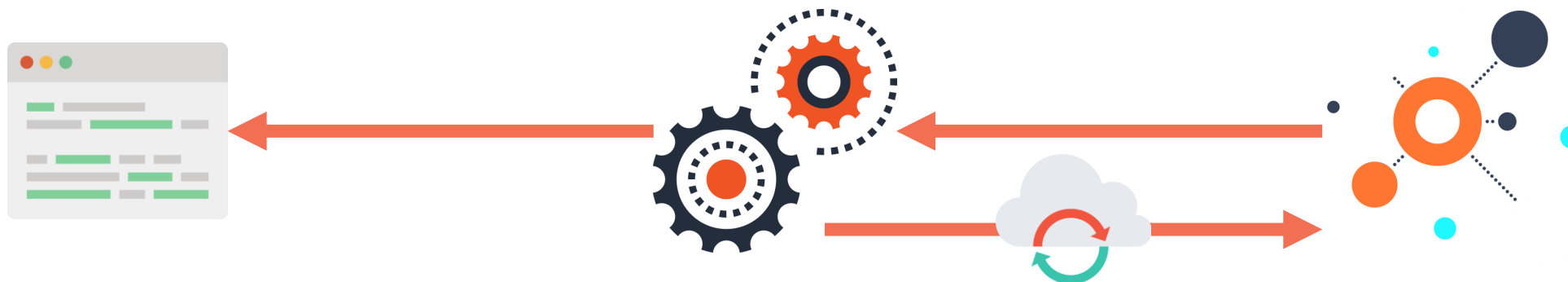


IndexedDB

an actual database in the browser

- Stores Data **Persistently**
- Stores **Structured** Data
- Supports **Range Queries**
- **Browser Support** 94%

Service Worker Background Sync



One-off Sync

- executed when user is **online**
- **retried** when failed (exponential backoff)

Use Cases

- Save **file** when online again
- Send **email** when online again

Periodic Sync

- executed when online, according to **period options**

Use Cases

- Load updates to **social media timeline** when browser closed

Service Worker Debugging



Chromium DevTools interface showing Service Worker debugging for **www.baqend.com**.

Service Workers Panel:

- Manifest: **Service Workers** (selected)
- Clear storage
- Storage: Local Storage, Session Storage, IndexedDB, Web SQL, Cookies
- Cache: Cache Storage, Application Cache
- Service Workers: **www.baqend.com** (Source: **sw.7dbf553e.js**, Received 23.1.2018, 15:38:40)
- Status: **#823 activated and is running** (stop)
- Clients: **https://www.baqend.com/** (focus)
- Push: Test push message from DevTools. (Push)
- Sync: test-tag-from-devtools (Sync)

Sources Panel:

- sw.7dbf553e.js:formatted (selected)
- Line 32, Column 9: `t.p = '';`

Service Worker **Caching**



Cache Storage

Stores Request/Response pairs

Cache Storage

- **Programmatically** managed
- **Persistent** and non-expiring
- Supports only **HTTP**
- Only caches **GET** requests (no HEAD)

Caching Strategies – Cache Only



Gets all requests from cache or fails.

Caching Strategies – Cache, Network Fallback



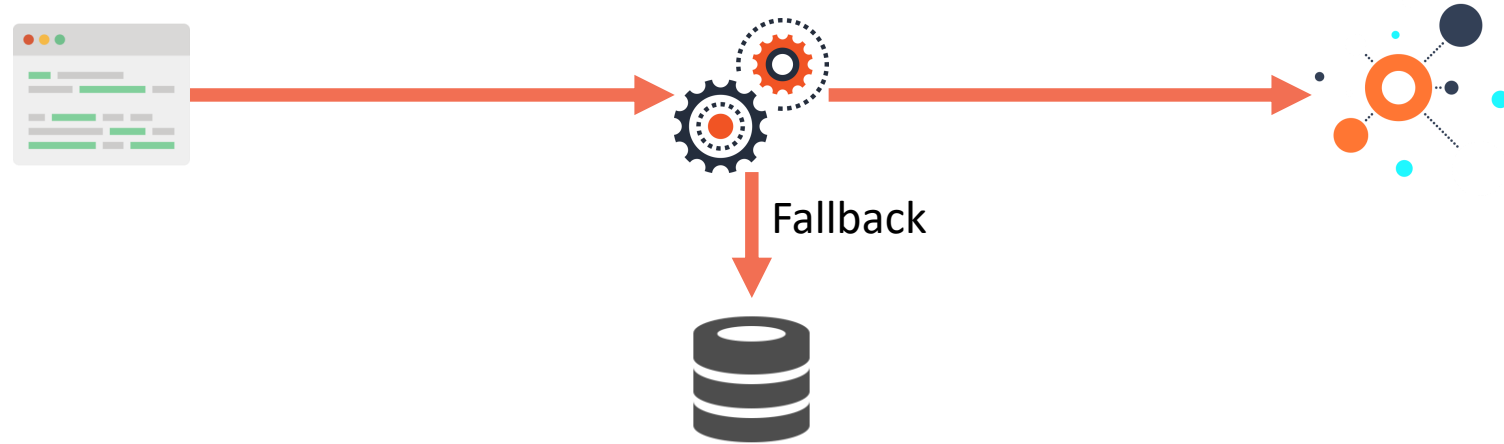
Gets requests from cache & uses network as fallback.

Caching Strategies – Network Only



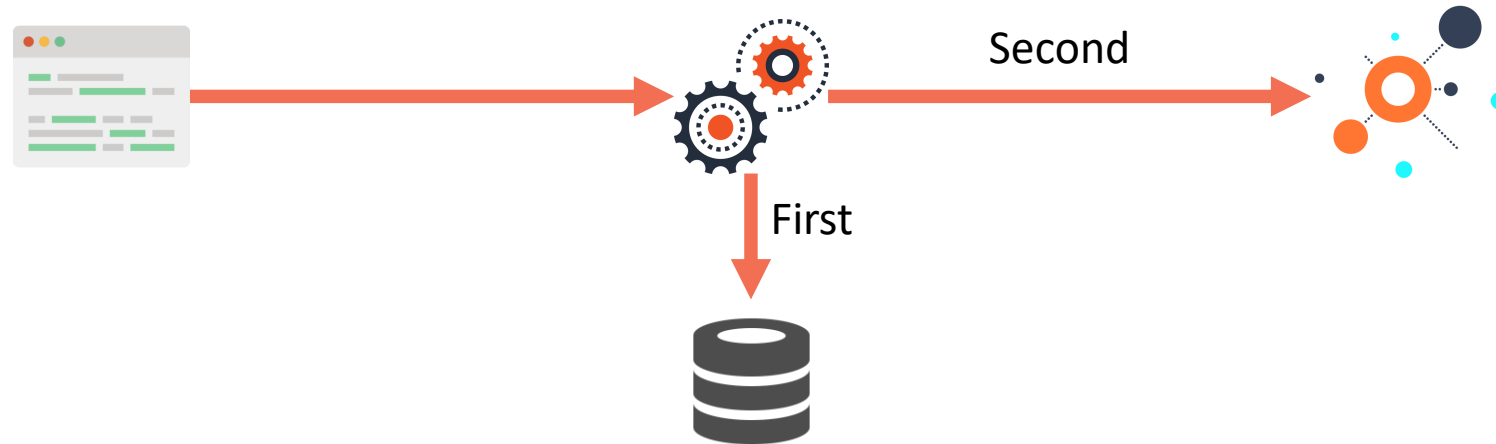
Gets requests from network only.

Caching Strategies – Network, Cache Fallback



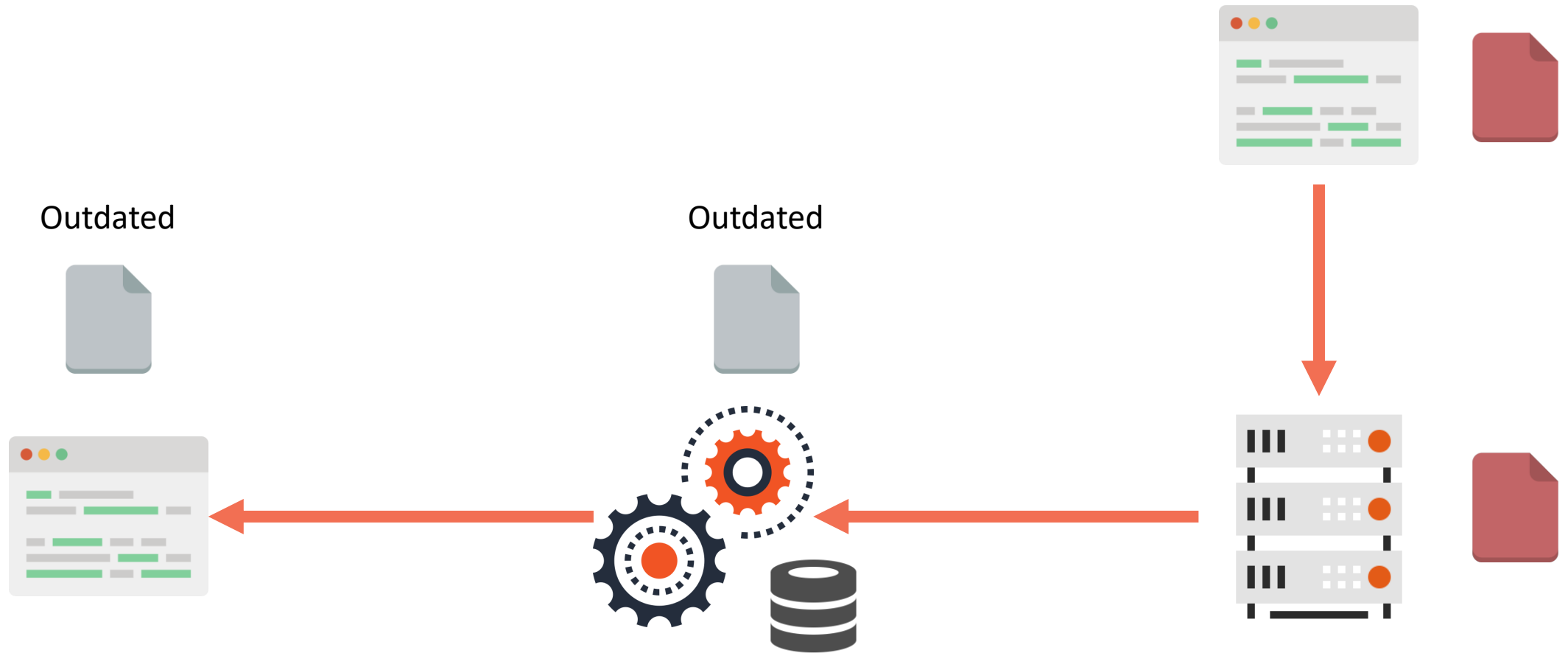
Gets requests from network, the cache acts as fallback (offline mode).

Caching Strategies – Cache, then Network



Gets requests from cache first and from network in background.

Major Challenge: Cache Coherence

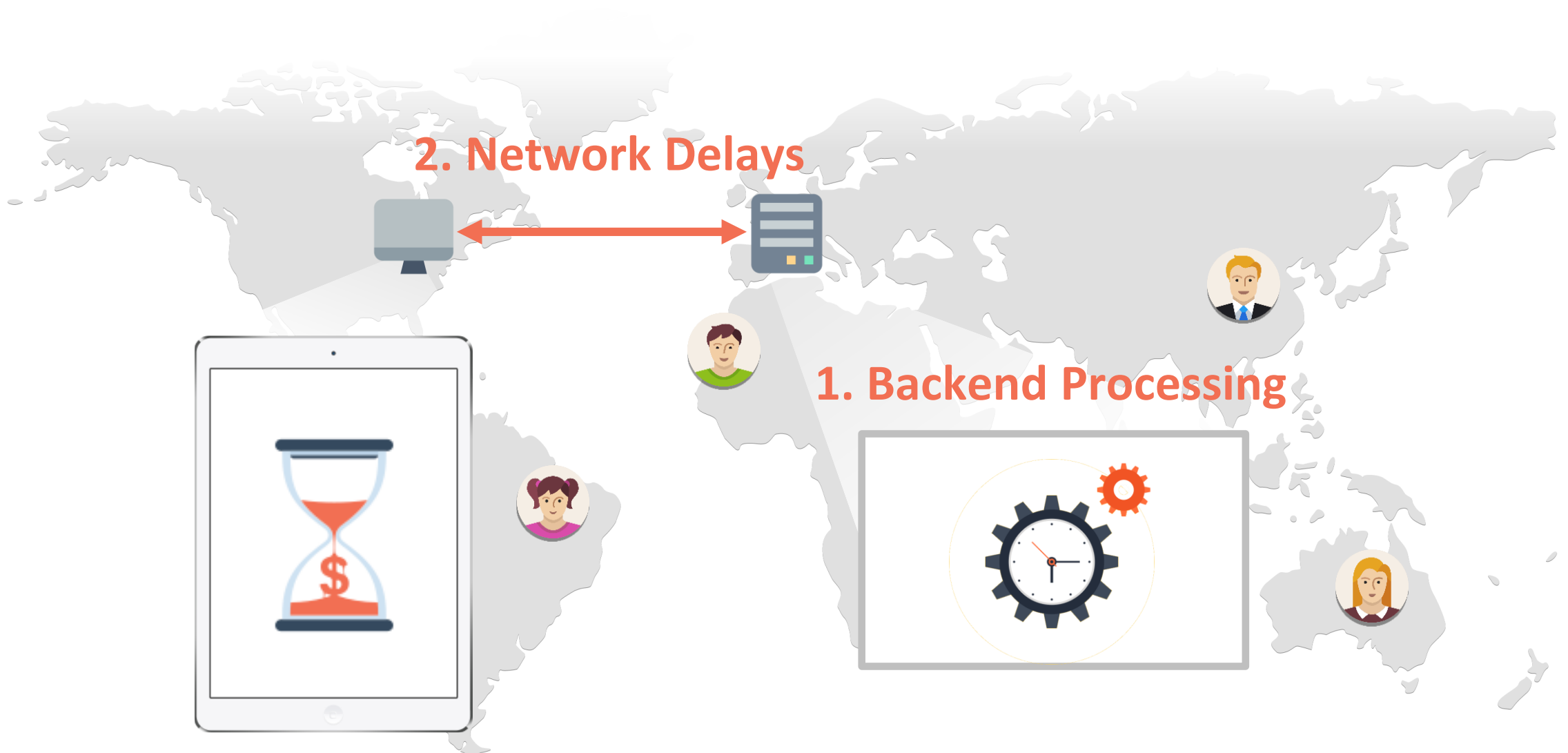


➡ All strategies either serve **outdated data** or **degrade performance**

A technician is working on a piece of electronic equipment. In the foreground, a Tektronix TDS 3054 oscilloscope displays two waveforms on its screen. To the right, a LEADER LV 5800 multimeter is being used, with its probes connected to a component. The background shows a color calibration chart and various cables and components on a workbench.

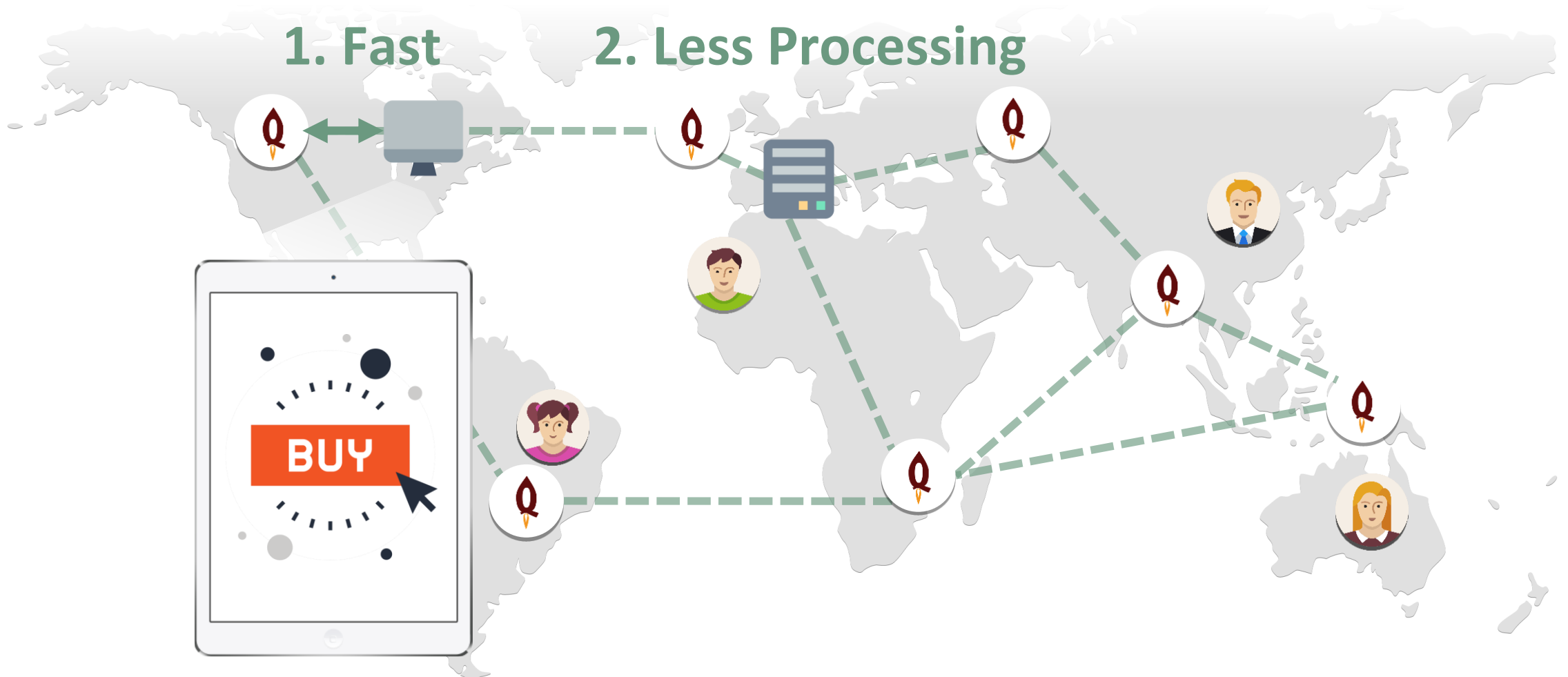
How we use Service Workers at Baqend

Problem: **slow backends & networks.**



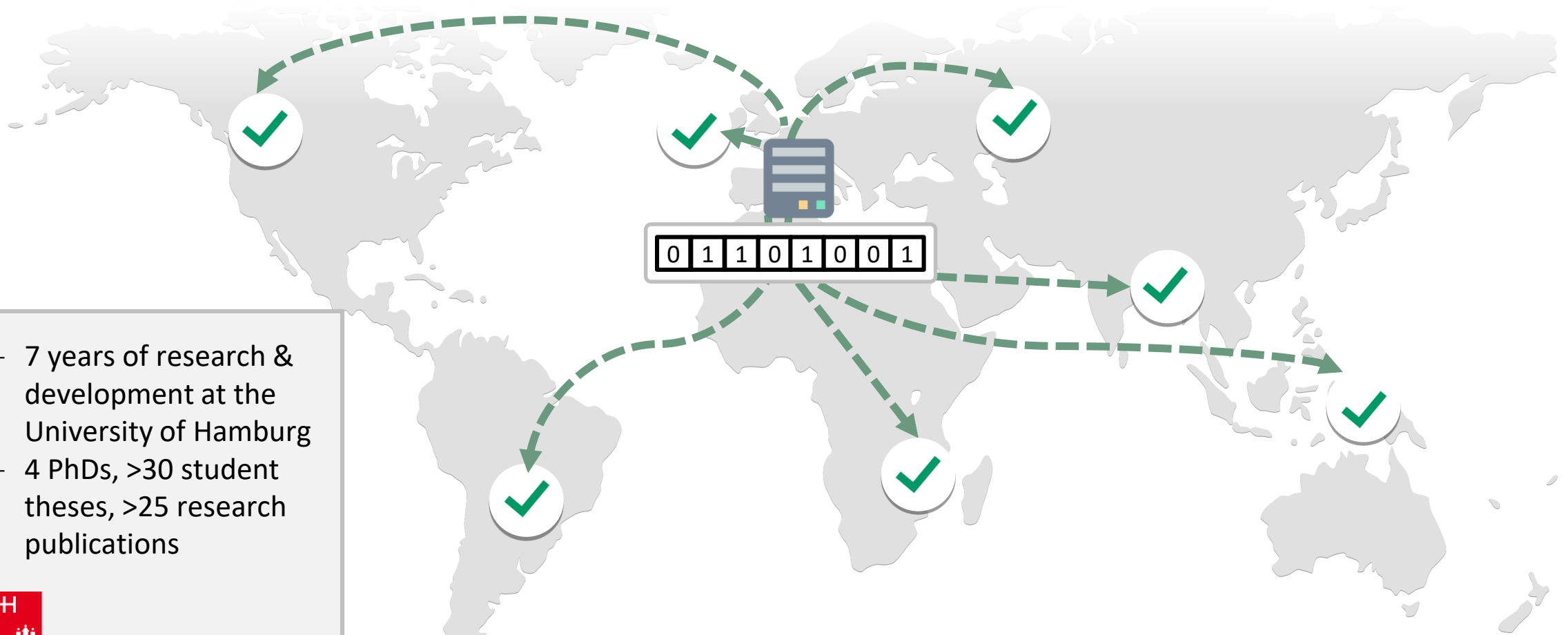
Solution: Speed Kit

Service Worker rewrites & accelerates slow requests.



The magic: dynamic data is kept **up-to-date**.

Backed by 30 man-years of **research**.



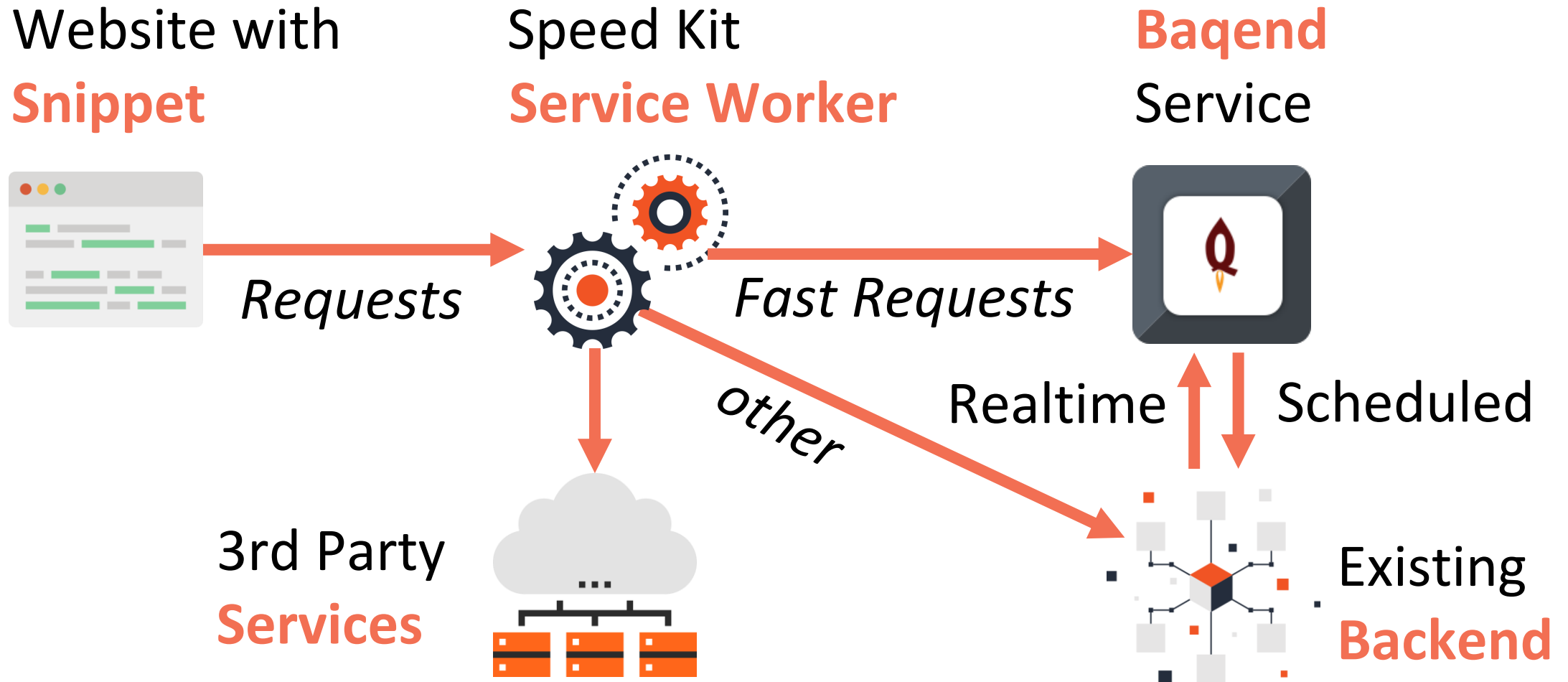
- 7 years of research & development at the University of Hamburg
- 4 PhDs, >30 student theses, >25 research publications



Universität Hamburg

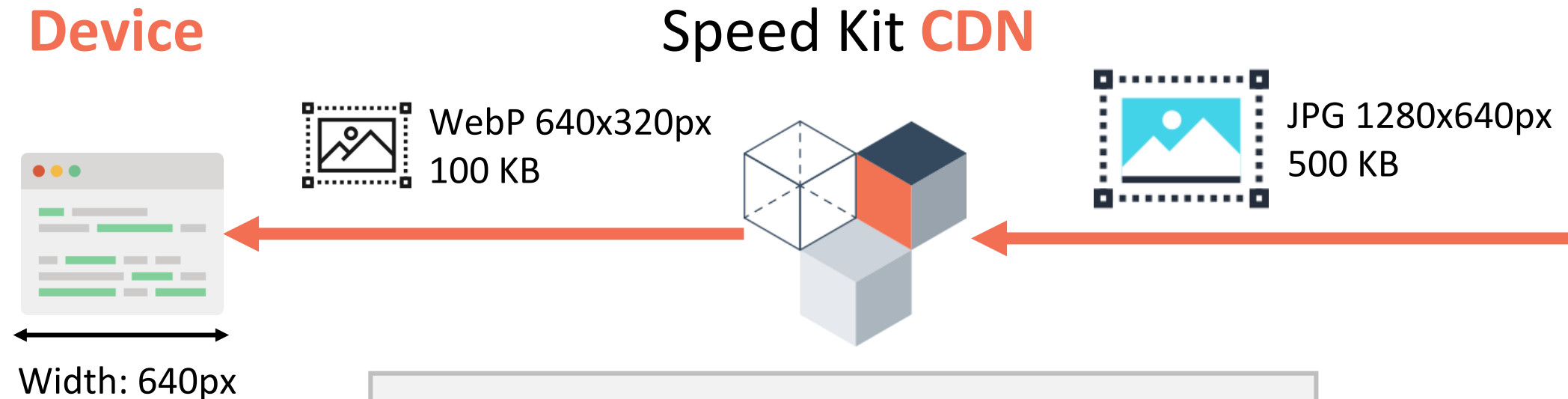
Learn [more](#).

How Speed Kit leverages **Service Workers**.



Use case I: optimize **images**.

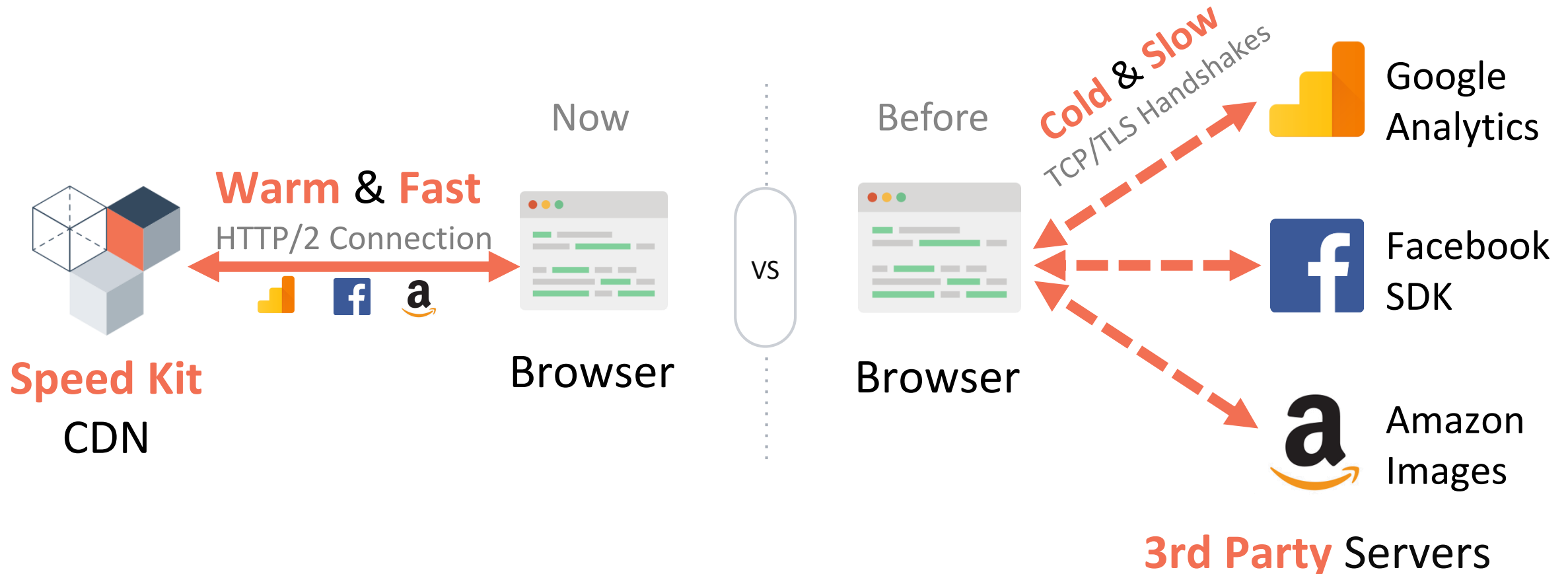
SW sends client resolution → responsive image.



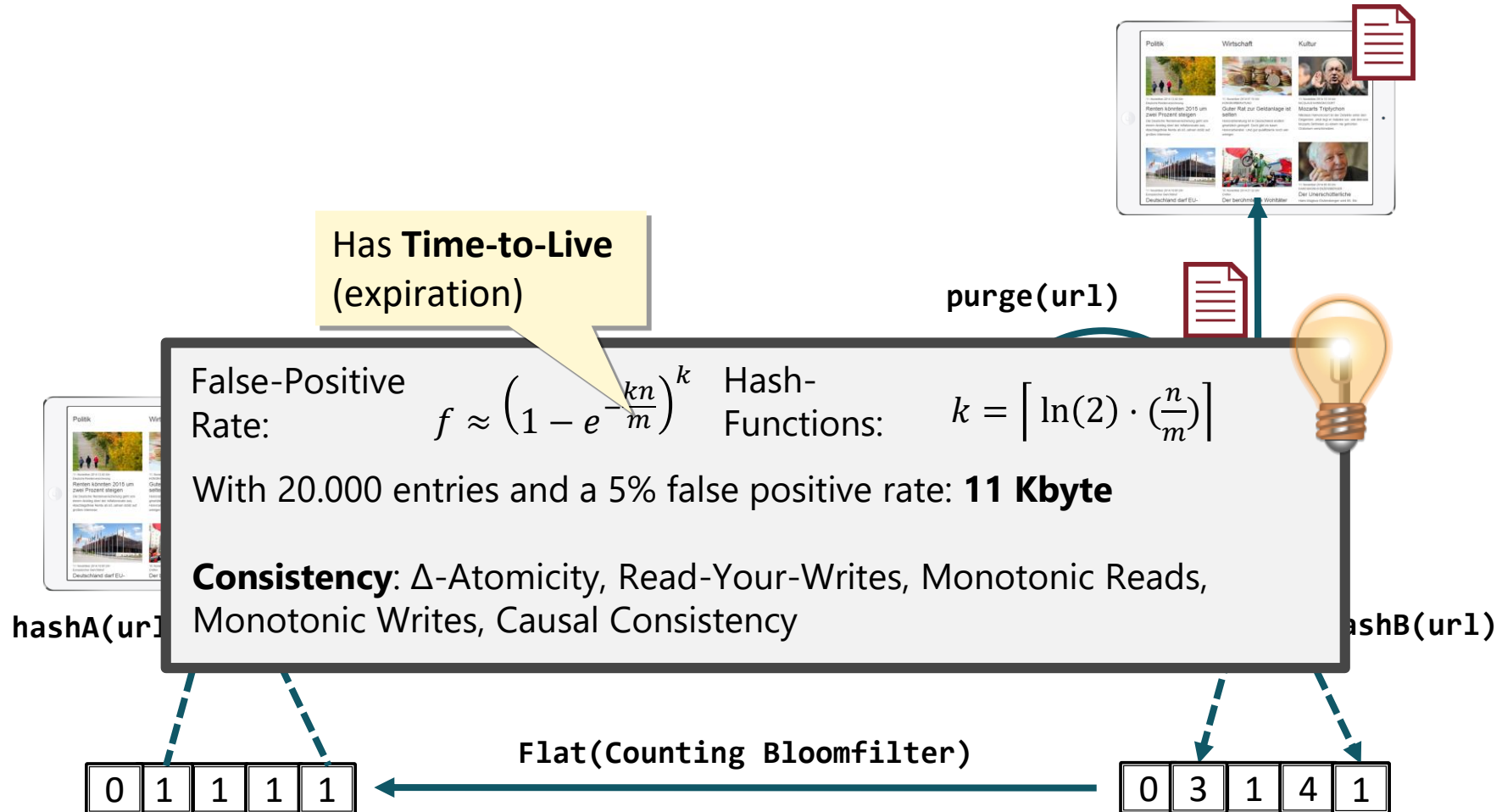
- ✓ Images transcoded to **WebP**
- ✓ Rescaled to match **Screen Size**
- ✓ JPG and PNG **Recompression**

Use case II: **re-route 3rd party dependencies.**

Service Workers can manipulate other domains.



Use case III: handling **cache coherence**.





Sign In For Web Push

Use case IV: simple **web push**.

A person stands in the center of a dark, cavernous space, holding a flashlight that casts a bright beam of light onto the ground. The cave walls are rugged and textured, with some greenish-yellow patches. The overall atmosphere is mysterious and exploratory.

Demo: Looking into Service Workers



Now, we have a
Progressive Web App.
**How do we measure
its performance?**

A **PWA** can make a huge difference.

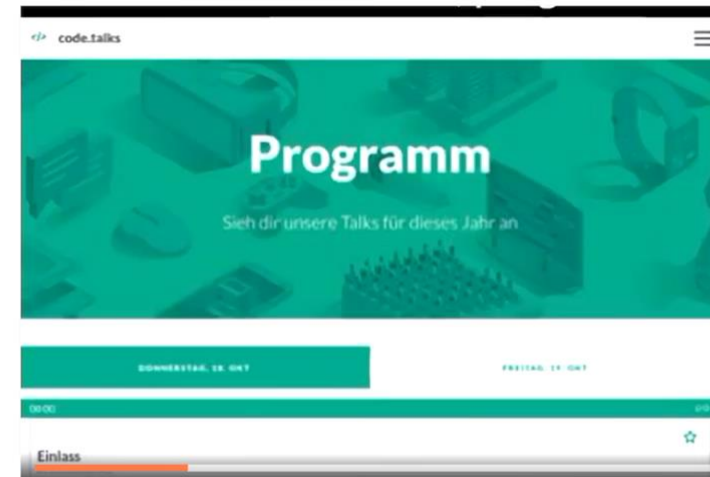
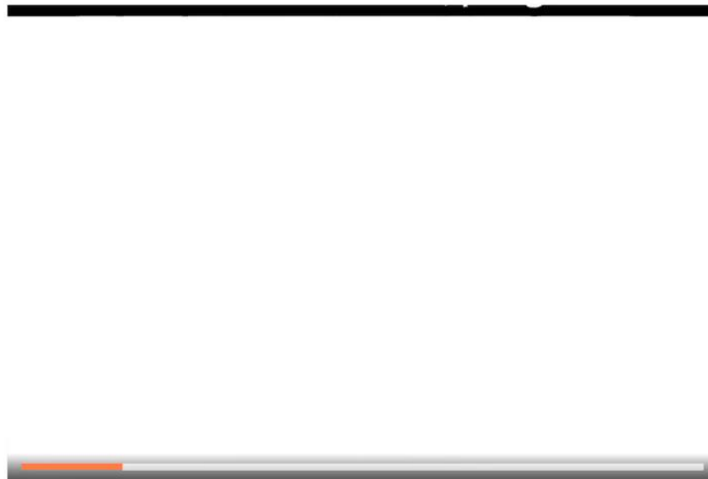
<https://www.codetalks.de/program>



Your Website
4300 ms

4.4x
Faster

With Speed Kit
980 ms



Your Website

4.3s

Average

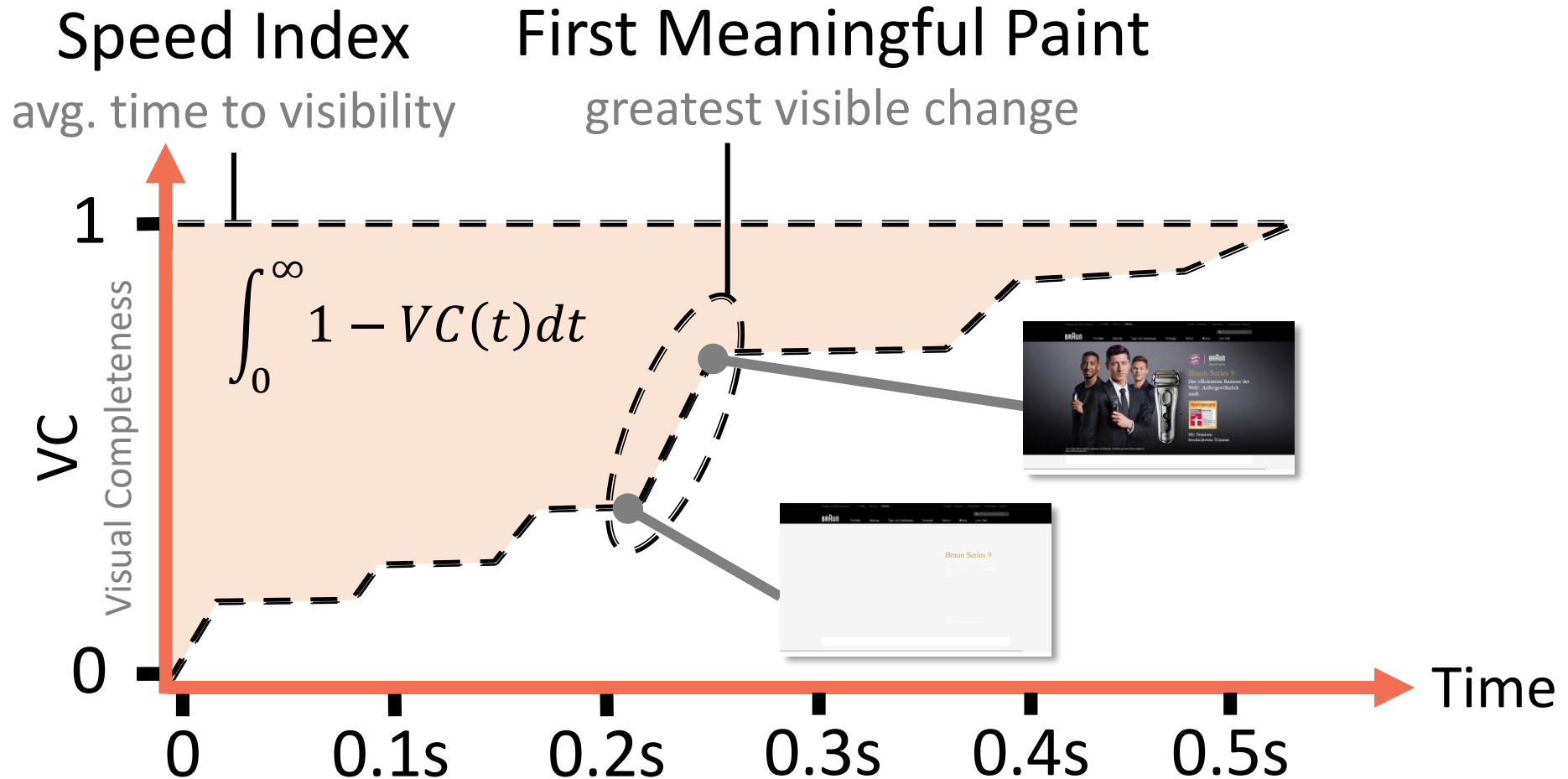
With Speed Kit

1s

Fast

Measuring PWA performance.

User-perceived performance.



Test **your** site.

www.example.com

Go

test.speed-kit.com

Wrap Up.

PWAs



Super cool
alternative
to native apps

Service Workers



Powerful
programmable
network proxy

Use Case



Speed Kit:
Smart CDN though
Service Workers



Edit 🔍



Learn more about this topic:

<https://blog.baqend.com/>

Applause from you, Konstantin Möllers, and 12 others



Wolfram Wingerath

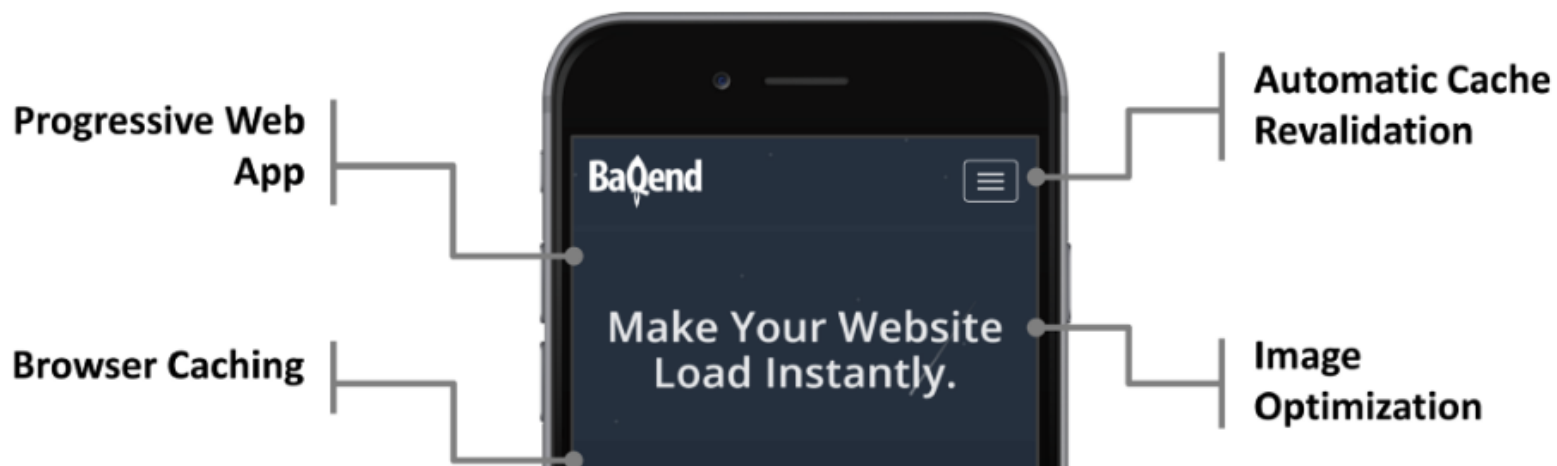
Distributed systems engineer at Baqend, a serverless backend for faster websites. Background in database research & developing Baqend's real-time query engine.

Apr 29 · 34 min read

Rethinking Web Performance with Service Workers

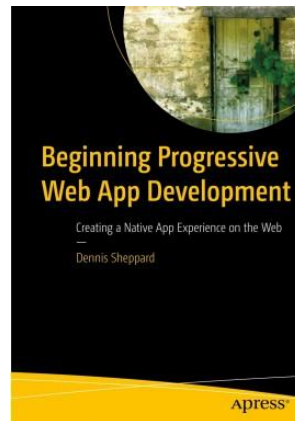
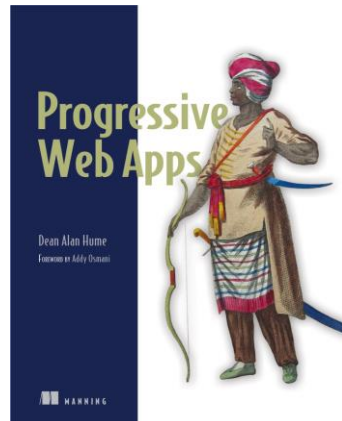
30 Man-Years of Research in a 30-Minute Read

This article surveys the current state of the art in page speed optimization. It contains the gist of more than 30 man-years of research that went into Speed Kit, an easy-to-use web performance plugin to accelerate any website.



Learn more about Services Workers.

Recommended Books



Blogs



Baqend Blog
On Building a Faster Web

<https://blog.baqend.com/>

Ilya Grigorik
Internet plumber

<https://www.igvita.com/>



<https://jakearchibald.com/>

Guides & Tutorials

Progressive Web Apps

A new way to deliver amazing user experiences on the web.

<https://developers.google.com/web/progressive-web-apps/>

Progressive web apps

Jump to: [PWA advantages](#) [Core PWA guides](#) [Technology guides](#)

<https://developer.mozilla.org/en-US/docs/Web/Apps/Progressive>

Catch our **other** talks!

14:00 - Kino 7 - Buzzing Technologies

Creating High-Performance Web Apps
with WebAssembly

15:00 - Kino 6 - Architecture

Real-Time Processing Explained: A
Survey of Storm, Samza, Spark & Flink