The Technology Behind Progressive Web Apps
Service Workers in Detail

Erik Witt
ew@baqend.com
September 12, techcamp 2018, Hamburg
@baqendcom
What we are going to cover.

- **PWA**
  - Core Features
  - Building Blocks
  - Implementation

- **Service Workers**
  - Lifecycle
  - Network Interception
  - Caching Strategies

- **Speed Kit**
  - Cache Coherence
  - Performance Measures
Why do(n’t) we love native apps?

Progressive Web Apps

seek to combine the great from native and web apps
What are Progressive Web Apps?
Progressive Web Apps (PWAs)

Fast Loads through Caching + Offline Mode (Synchronization) + Add-to-Homescreen and Push Notifications
Try this:

www.baqend.com
Building Blocks of PWAs

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

- **1. Manifest**
- **2. Service Worker**

Progressively enhance when supported.
Implementing PWAs

PWAs are best practices and open web standards

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

```html
<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**:

![Diagram showing Web Push and Background Sync](image-url)
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 of Progressive Web Apps?
The Future of **PWAs** is bright.

**Payment Request API**

- **Goal:** replace traditional **checkout** forms
- **Just ~10 LOC** to implement payment
- **Vendor- & Browser-Agnostic**
The Future of PWAs is bright.

Credentials Management API

1. Click **Sign-in** → Native Account Chooser

2. Credentials API **stores** information for future use

3. **Automatic** Sign-in afterwards
The Future of **PWAs** is bright.

**Geofencing**

- **Notify** web app when user leaves or enters a defined area
- Requires **permission**
The Future of **PWAs** is bright.

**Web Speech API**

Native Speech Recognition in the Browser:

```javascript
annyang.addCommands({
  'Hello Meetup': () => {
    console.log('Hello you. ');
  }
});
```
The Future of PWAs is bright.

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**

<table>
<thead>
<tr>
<th>Browser</th>
<th>Support Status</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Edge</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Firefox</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Chrome</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Safari</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>iOS Safari</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Opera Mini</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Chrome for Android</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>UC Browser for Android</td>
<td>Supported</td>
<td>~90%</td>
</tr>
<tr>
<td>Samsung Internet</td>
<td>Supported</td>
<td>~90%</td>
</tr>
</tbody>
</table>

Supported by ~90% of browsers. Requires **TLS Encryption**.
Late, but all in: **Microsoft**

Publish PWAs to Microsoft Store

or

Bing Crawls PWAs → Convert to AppX → Microsoft Store

[https://blogs.windows.com/msedgedev/2018/02/06/welcoming-progressive-web-apps-edge-windows-10/#tqlAYGJrOUcxvCWg.97](https://blogs.windows.com/msedgedev/2018/02/06/welcoming-progressive-web-apps-edge-windows-10/#tqlAYGJrOUcxvCWg.97)
How are **Service Workers** registered?

```html
<script>
    navigator.serviceWorker.register('/sw.js');
</script>
```
How does the **Lifecycle** look like?

```javascript
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?

How to communicate with service workers involves sending messages to the browser tab and receiving push notifications. Here's how you can do it:

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

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

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

```javascript
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** Request
- **Change** Response
- **Concat** Responses
- **Cache** Responses
- **Serve** Cached Data
- ...
Service Worker Scope

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

Scope determines which requests go to the Service Worker
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

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

IndexedDB
an actual database in the browser
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

*Experimental*
Service Worker Debugging
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.
What we do with Service Workers
Speed Kit

Turning Websites into Instantly-Loading Progressive Web Apps
What **Speed Kit** does.
What Speed Kit does.
What Speed Kit does.

Backed by 30 Man-Years of Research
How Speed Kit solves Cache Coherence

Has Time-to-Live (expiration)

False-Positive Rate: \( f \approx \left(1 - e^{-\frac{kn}{m}}\right)^k \)

Hash-Functions: \( k = \left\lfloor \ln(2) \cdot \left(\frac{n}{m}\right) \right\rfloor \)

With 20,000 entries and a 5% false positive rate: 11 Kbyte

Consistency: \( \Delta \)-Atomicity, Read-Your-Writes, Monotonic Reads, Monotonic Writes, Causal Consistency

Flat (Counting Bloomfilter)

0 1 1 1 1

0 3 1 4 1
Adding **Speed Kit** to a Site
1. Configure Domain

Set which sites/URLS Baqend should accelerate (white- and blacklist, dynamic blocks).
2. Include Code Snippet

Add Speed Kit to your website’s HTML and upload the Service Worker.
3. Speed Kit is Active

Speed Kit intercepts requests and serves them through Baqend’s infrastructure.
How **it works** under the hood

Website with Snippet

Requests

3rd Party Services

Speed Kit Service Worker

Fast Requests

other

Realtime

Scheduled

Baqend Service

Existing Backend
Optimized & Cached Images

Device

WebP 640x320px 100 KB

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

Speed Kit CDN

JPG 1280x640px 500 KB

Width: 640px
Now, we have a Progressive Web App. How do we measure web performance?
Your Website
1131ms

3.1x Faster

With Speed Kit
366ms

With Speed Kit
0.4s

Excellent

Your Website
1.1s

Poor

Show Details
User-perceived performance

- **Speed Index**: average time to visibility
- **First Meaningful Paint**: greatest visible change

\[ \int_0^\infty 1 - VC(t) dt \]
Does it work for Your Site?

www.example.com

Go

test.speed-kit.com
Wrap Up

PWA
Super cool alternative to native apps

Service Workers
Powerful programmable network proxy

Speed Kit
Combines Service Workers & cache coherence
Learn more about this topic:

https://blog.baqend.com/

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.
Web Performance Literature

**Good Resources**

- [High Performance Mobile Web](https://hpbn.co/)
- [Building Progressive Web Apps](https://developers.google.com/speed/pagespeed/
- [Performance Tools](https://test.speed-kit.com)

**Google Developers Performance**

[https://developers.google.com/web/fundamentals/performance/?hl=en](https://developers.google.com/web/fundamentals/performance/?hl=en)

**Website Performance Optimization**


**Baqend Blog**

[https://medium.baqend.com/](https://medium.baqend.com/)

**Performance Tools**

- [PageSpeed Insights](https://developers.google.com/speed/pagespeed/)
- [Page Speed Analyzer](https://test.speed-kit.com)
- [Baqend](https://www.baqend.com/)
- [WebPageTest](http://www.webpagetest.org/)
Contact us.

Our Product

Speed Kit:
• Accelerates Any Website
• Pluggable
• Easy Setup

test.speed-kit.com

Our Services

• Web & Data Management Consulting
• Performance Auditing
• Implementation Services

Erik Witt
ew@baqend.com
Web Performance Engineer