Part II Chapter 11: PWA

Translator: Henry Lim 林亨力 (GitHub: @limhenry, Twitter: @henrylim96)
Last updated: July 30, 2020, 10:30 PM PDT

Introduction

渐进式 Web 应用(PWA)是一种新的Web应用程序，构建在平台原语(如Service Worker API)之上。Service Worker通过充当网络代理，拦截web应用程序发出的请求，并用编程或缓存的响应进行应答，从而允许应用程序支持独立于网络的加载。 Service worker可以用来接收推送，也可以在应用程序未开启的情况下进行后台同步。此外，通过 Web App Manifests，Service worker也能让PWA安装到用户设备的主屏幕上。

在2014年的十二月，Chrome 40支持了 Service worker。而在2015年，Frances Berriman 和 Alex Russell 创造了渐进式 Web 应用（Progressive Web Apps) 一词。到了今天，市面上所有的主流浏览器也支持了service worker，而在本章中将会提到在市面上到底有多少的PWA以及那些PWA如何使用这些新科技。一些比较进阶的API (例如Background Sync)，目前只能在基于Chromium的浏览器中使用，so as an additional question, we looked into which features these PWAs actually use.

Service Workers

Service worker 注册与安装

我们第一个要先探索的指标是Service worker安装。通过HTTP Archive的功能计数器，我们发现到0.44% 的桌面网页和0.37% 的移动网页注册了Service worker，而这两个的数目还在持续着快速增长。

虽然这数目看起来不怎么样，但是如果我们从Chrome Platform Status的流量数据，我们可以发现service worker在所有网页中控制了15%的网页加载。换句话说，越来越多的高流量网站已经开始使用service worker。

Lighthouse会检查该网页是否符合安装提示的条件。1.56%的移动网页符合了安装的条件。

0.82% 的桌面网页和 0.94%的移动网页使用了OnBeforeInstallPrompt接口来控制安装的体验。目前只有在基于Chromium的浏览器支持了OnBeforeInstallPrompt。

Service worker 事件

Service worker可以监听以下的事件:

- install, 当service worker被安装时触发。
- activate, 当service worker被激活时触发。
- fetch, 当页面发起网络请求时触发。
- push, 当接收到推送时触发。
- notificationclick, 当推送被点击时触发。
- notificationclose, 当推送被关闭时触发。
- message, 当接收到一个由postMessage()发送的信息时触发。
- sync, 当后台同步事件发生时触发。

We have examined which of these events are being listened to by service workers we could find in the HTTP Archive
我們已從 HTTP Archive 中 service worker
我们发现到fetch，install和 activate 事件在桌面网页和移动网页都是一樣常見，而這也是目前最常见的service worker事件，其次是 notificationclick 和 push 事件。从这结果看，我们发现开发者使用service worker 的最大原因是要把网页也能在离线下操作。而第二个使用service worker的原因是用来接收推送，但是这并不是特别的热门。由于后台同步只被少数的浏览器支持，再加上并没有很多的常见用例，我们发现后台同步取代扮演重要角色。

Service worker文件大小

一般来说，文件大小和代码行并不是一个很好来计算任务的复杂性。但是，在这情况下，比较在桌面网页和移动网页运行的 (已压缩的) service worker文件大小还是挺有趣的。

在桌面网页中，service worker的中位数大小是895字节，而在移动网页则是694字节。这代表在桌面网页中service worker是比在移动网页还要大的。这些统计信息并未包括通过 importScripts() 方法动态加载的脚本。如果统计信息也包括动态加载的脚本的话，这可能会使service worker的文件大小更大。

Web app manifest

Web app manifest 屬性

The web app manifest is a simple JSON file that tells the browser about a web application and how it should behave when installed on the user's mobile device or desktop. A typical manifest file includes information about the app name, icons it should use, the start URL it should open at when launched, and more. Only 1.54% of all encountered manifests were invalid JSON, and the rest parsed correctly.

We looked at the different properties defined by the Web App Manifest specification, and also considered non-standard proprietary properties. According to the spec, the following properties are allowed:

Lighthouse要求至少包含一个192x192像素的图标，但是市面上的图标生成工具也会创建其他尺寸的图标。

尽管谷歌的文档明确建议使用512x512像素的图标，但Lighthouse的图标像素要求可能是导致192x192像素的图标成为台式机和移动设备上图标尺寸最受欢迎的原因。

Workbox

Workbox是一组库，可帮助解决常见的 service worker 的用例。比如說，Workbox可以在您的构建过程中創建需要预缓存文件的列表。Workbox包含用于处理运行时缓存，请求路由，缓存过期，后台同步等的库。

鉴于Service Worker API的低级性质，许多开发人员已將Workbox用作将其service worker的逻辑构造为更高级别的，可重用代码块的一种方式。

Given the low-level nature of the service worker APIs, many developers have turned to Workbox as a way of structuring their service worker logic into higher-level, reusable chunks of code. Workbox adoption is also driven by its inclusion as a feature in a number of popular JavaScript framework starter kits, like create-react-app and Vue's PWA plugin.

The HTTP Archive shows that 12.71% of websites that register a service worker are using at least one of the Workbox libraries. This percentage is roughly consistent across desktop and mobile, with a slightly lower percentage (11.46%) on mobile compared to desktop (14.36%).

结论

从本章的统计看来，PWA只是占了市面上网页的一小部分。但是，这较小的流量却是来自一些高流量的热门网页。我们也发现了15%的网页加载使用了service worker。Service worker 也带来了一系列的好处，例如说这将会提高网页的性能，这也能让移动设备可以更好地控制缓存。这也意味service worker 的使用量将会继续增长。

PWAs have often been seen as Chrome-driven technology. Other browsers have made great strides recently to implement most of the underlying technologies, although first-class installability lags on some platforms. It's positive to see support becoming more widespread. Maximiliano Firtman does a great job of tracking this on iOS, including explaining Safari PWA support. Apple doesn't use the term PWA much, and has explicitly stated that these HTML5 apps are best delivered outside of the App Store. Microsoft went the opposite direction, not only encouraging PWAs in its app store, but even automatically shortlisting PWAs to be added that were found via the

Bing web crawler. Google has also provided a method for listing web apps in the Google Play Store, via Trusted Web Activities.

PWAs provide a path forward for developers who would prefer to build and release on the web instead of on native platforms and app stores. Not every operating system and browser offers full parity with native software, but improvements continue, and perhaps 2020 is the year where we see an explosion in deployments?