Progressive web app

A **progressive web application** (**PWA**), or **progressive web app**, is a type of web app that can be installed on a device as a standalone application.[1] PWAs are installed using the offline cache of the device's web browser.[2]

PWA logo

PWAs were introduced from 2016 as an alternative to native (device-specific) applications, with the advantage that they do not require separate bundling or distribution for different platforms. They can be used on a range of different systems, including desktop and mobile devices. Publishing the app to digital distribution systems, such as the Apple App Store, Google Play, or the Microsoft Store on Windows, is optional.[2]

Because a PWA is delivered in the form of a webpage or website built using common web technologies including HTML, CSS, JavaScript, and WebAssembly,[3] it can work on any platform with a PWA-compatible browser. As of 2025, PWA features are supported to varying degrees by Google Chrome, Apple Safari, Brave, Firefox for Android, and Microsoft Edge[4][5] but not by Firefox for desktop.[6]

History

Predecessors

At Apple's Worldwide Developers Conference in 2007, Steve Jobs announced that the iPhone would "run applications created with Web 2.0 Internet standards".[7] No software development kit (SDK) was required, and the apps would be fully integrated into the device through the Safari browser engine.[8] This model was later switched to the App Store, as a means of appeasing frustrated developers.[9] In October 2007 Jobs announced that an SDK would be launched the following year.[8] As a result, although Apple continued to support web apps, the vast majority of iOS applications shifted toward the App Store.

Beginning in the early 2010s dynamic web pages allowed web technologies to be used to create interactive web applications. Responsive web design, and the screen-size flexibility it provides have made PWA development more accessible. Continued enhancements to HTML, CSS, and JavaScript allowed web applications to incorporate greater levels of interactivity, making native-like experiences possible on a website.[10]

In 2013, Mozilla released Firefox OS. It was intended to be an open-source operating system for running web apps as native apps on mobile devices. Firefox OS was based on the Gecko rendering engine with a user interface called Gaia, written in HTML5. The development of Firefox OS ended in 2016,[11] and the project was completely discontinued in 2017,[12] although a fork of Firefox OS was used as the basis of KaiOS, a feature phone platform.[13]

Initial introduction

In 2015, designer Frances Berriman and Google Chrome engineer Alex Russell coined the term "progressive web apps"[14] to describe apps taking advantage of new features supported by modern browsers, including service workers and web app manifests, that let users upgrade web apps to progressive web applications in their native operating system (OS). Google then put significant efforts into promoting PWA development for Android.[15][16] Firefox introduced support for service workers in 2016, and Microsoft Edge and Apple Safari followed in 2018,[17][15] making service workers available on all major systems.

By 2019, PWAs were supported by desktop versions of some browsers, including Microsoft Edge[18] (on Windows) and Google Chrome[19] (on Windows, macOS, ChromeOS, and Linux).

In December 2020, Firefox for desktop abandoned the implementation of PWAs (specifically, removed the prototype "site-specific browser" configuration that had been available as an experimental feature). A Firefox architect noted: "The signal I hope we are sending is that PWA support is not coming to desktop Firefox anytime soon."[6] Mozilla supports PWAs on Android and plans to keep supporting it.[20]

Browser support

Browser	Support					Comment
	Windows	macOS	Linux	Android	iOS & iPadOS	
Chromium-based	Yes	Yes	Yes	Yes	—	Includes Google Chrome, Microsoft Edge,[18] Brave, Opera, Vivaldi,[21] and others.
Firefox	No [6]	No [6]	No [6]	Partial	No	
Safari	—	Partial [22]	—	—	Partial [23]	

Characteristics

Progressive web apps are all designed to work on any browser that is compliant with the appropriate web standards. As with other cross-platform solutions, the goal is to help developers build cross-platform apps more easily than they would with native apps.[15] Progressive web apps employ the progressive enhancement web development strategy.

Some progressive web apps use an architectural approach called the App Shell Model.[24] In this model, service workers store the Basic User Interface or "shell" of the responsive web design web application in the browser's offline cache. This model allows for PWAs to maintain native-like use with or without web connectivity. This can improve loading time, by providing an *initial static frame*, a layout or architecture into which content can be loaded progressively as well as dynamically.[25]

Technical criteria

The technical baseline criteria for a site to be considered a progressive web app and therefore capable of being installed by browsers were described by Russell in 2016[26] and updated since:[27][28]

- Originate from a secure origin. Served over TLS and have no active mixed content. Progressive web apps must be served via HTTPS to ensure user privacy, security, and content authenticity.
- Register a service worker with a fetch handler. Progressive web apps must use service workers to create programmable content caches. Unlike regular HTTP web cache, which caches content after the first use and then rely on various heuristics to guess when content is no longer needed, programmable caches can explicitly prefetch content in advance *before* it's used for the first time and explicitly discard it when it is no longer needed.[29] This requirement helps pages to be accessible offline or on low-quality networks.
- Reference a web app manifest. The manifest must contain at least the five key properties: `name` or `short_name`, `start_url`, and `display` (with a value of `standalone`, `fullscreen` or `minimal-ui`), and `icons` (with 192 px and a 512 px versions). Information contained in the manifest makes PWAs easily shareable via a URL, discoverable by search engines, and alleviates complex installation procedures (but PWAs may still be listed in a third-party app store).[30] Furthermore, PWAs support native app-style interactions and navigation, including being added to the home screen, displaying splash screens, etc.

Technologies

There are many technologies commonly used to create progressive web apps. A web application is considered a PWA if it satisfies the installation criteria, thus can work offline and can be added to the device's home screen. To meet this definition, all PWAs require at minimum a manifest and a service worker.[31][32][33] Other technologies may be used to store data, communicate with servers or execute code.

Manifest

The web app manifest[34] is a World Wide Web Consortium (W3C) specification defining a JSON-based manifest (usually labelled manifest.json)[30] to provide developers with a centralized place to put metadata associated with a web application including:

- The name of the web application
- Links to the web app icons or image objects
- The preferred URL to launch or open the web app
- The web app configuration data
- Default orientation of the web app
- The option to set the display mode, e.g. full screen

This metadata is crucial for an app to be added to a home screen or otherwise listed alongside native apps.

iOS support

iOS Safari partially implements manifests, while most of the PWA metadata can be defined via Apple-specific extensions to the meta tags. These tags allow developers to enable full-screen display, define icons and splash screens, and specify a name for the application.[35][36]

Service workers

A service worker is a web worker that implements a programmable network proxy that can respond to web/HTTP requests from the main document. It is able to check the availability of a remote server, cache content when that server is available, and serve that content to the document later. Service workers, like any other web workers, work separately from the main document context. Service workers can handle push notifications and synchronize data in the background, cache or retrieve resource requests, intercept network requests and receive centralized updates independently of the document that registered them, even when that document is not loaded.[37]

Service workers go through a three-step lifecycle of Registration, Installation and Activation. Registration involves telling the browser the location of the service worker in preparation for installation. Installation occurs when there is no service worker installed in the browser for the web app, or if there is an update to the service worker. Activation occurs when all of the PWA's pages are closed, so that there is no conflict between the previous version and the updated one. The lifecycle also helps maintain consistency when switching among versions of a service worker since only a single service worker can be active for a domain.[37]

WebAssembly

WebAssembly allows precompiled code to run in a web browser, at near-native speed.[38] Thus, libraries written in languages such as C can be added to web apps. Announced in 2015 and first released in March 2017, WebAssembly became a W3C recommendation on December 5, 2019[39][40] and it received the Programming Languages Software Award from ACM SIGPLAN in 2021.[41]

Data storage

Progressive web app execution contexts get unloaded whenever possible, so progressive web apps need to store the majority of their long-term internal state (user data, dynamically loaded application resources) in one of the following manners:

Web storage

Web storage is a W3C standard API that enables key–value storage in modern browsers. The API consists of two objects, `sessionStorage` (that enables session-only storage that gets wiped upon browser session end) and `localStorage` (that enables storage that persists across sessions).[42]

Indexed Database API

Indexed Database API is a W3C standard database API available in all major browsers. The API is supported by modern browsers and enables storage of JSON objects and any structures representable as a string.[43] The Indexed Database API can be used with a wrapper library providing additional constructs around it.

Comparison with native apps

In 2017, Twitter released Twitter Lite, a PWA alternative to the official native Android and iOS apps. According to Twitter, Twitter Lite consumed only 1–3% of the size of the native apps.[44] Starbucks provides a PWA that is 99.84% smaller than its equivalent iOS app. After deploying its PWA, Starbucks doubled the number of online orders, with desktop users ordering at about the same rate as mobile app users.[45]

A 2018 review published by Forbes, found that users of Pinterest's PWA spent 40% more time on the site compared to the previous mobile website. Ad revenue rates also increased by 44%, and core engagements by 60%.[46] Flipkart saw 60% of customers who had uninstalled their native app return to use the Flipkart PWA. Lancôme saw an 84% decrease in time until the page is interactive, leading to a 17% increase in conversions and a 53% increase in mobile sessions on iOS with their PWA.[47]

Release via app stores

Since a PWA does not require separate bundling or distribution for different platforms and is available to users via the web, it is not necessary for developers to release it over digital distribution systems like the Apple App Store, Google Play, Microsoft Store, or Samsung Galaxy Store. The major app stores support the publication of PWAs to varying to degrees.[2] Google Play, Microsoft Store,[48] and Samsung Galaxy Store support PWAs, but Apple App Store does not. Microsoft Store publishes some qualifying PWAs automatically (even without app authors' requests) after discovering them via Bing indexing.[49]

See also

- Google Lighthouse, an open-source audit tool for PWAs developed by Google

References

1. "What are Progressive Web Apps? PWA Guide for Beginners" (https://www.freecodecamp.org/news/what-are-progressive-web-apps-pwa-guide/). *freeCodeCamp.org*. 2024-01-18. Retrieved 2024-05-06.
2. "Progressive Web Apps | Software AG" (https://techradar.softwareag.com/technology/progressive-web-apps/). *techradar.softwareag.com*. Retrieved 2020-09-25.
3. Ltd, Cybellium. *Mastering Front-end development* (https://books.google.com/books?id=xbbnEAAAQBAJ&dq=Progressive+web+app+is+intended+to+work+on+any+platform+with+a+standards-compliant+browser,+including+desktop+and+mobile+devices.&pg=PA273). Cybellium Ltd. p. 273. ISBN 979-8-8668-4882-9.
4. "Can I use pwa?" (https://caniuse.com/#search=pwa). *CanIUse*. Retrieved 27 January 2021.
5. "Is Service Worker Ready?" (https://jakearchibald.github.io/isserviceworkerready/). Jake Archibald.
6. Newman, Jared (2021-01-26). "Firefox just walked away from a key piece of the open web" (https://www.fastcompany.com/90597411/mozilla-firefox-no-ssb-pwa-support). *Fast Company*. Retrieved 2021-01-27.

7. Jobs, Steve; Apple (11 June 2007). "iPhone to Support Third-Party Web 2.0 Applications" (https://www.apple.com/newsroom/2007/06/11iPhone-to-Support-Third-Party-Web-2-0-Applications/). *Apple*.
8. Ritchie, Rene (5 March 2018). "App Store Year Zero: Unsweet web apps drove iPhone to an SDK" (https://www.imore.com/history-app-store-year-zero). *iMore*. Retrieved 23 May 2019.
9. "Jobs' original vision for the iPhone: No third-party native apps" (https://9to5mac.com/2011/10/21/jobs-original-vision-for-the-iphone-no-third-party-native-apps/). *9to5Mac*. 21 October 2011. Retrieved 22 May 2019.
10. Marcotte, Ethan (25 May 2010). "Responsive Web Design" (http://alistapart.com/article/responsive-web-design). *A List Apart*. Retrieved May 25, 2010.
11. "Mozilla ends commercial Firefox OS development - gHacks Tech News" (https://www.ghacks.net/2016/09/27/mozilla-ends-commercial-firefox-os-experiment/). *gHacks Technology News*. 2016-09-27. Retrieved 2022-05-05.
12. Hoffman, Chris; PCWorld | (2016-09-28). "Mozilla is stopping all commercial development on Firefox OS" (https://www.pcworld.com/article/3124563/mozilla-is-stopping-all-commercial-development-on-firefox-os.html). *PCWorld*. Retrieved 2021-03-17.
13. "KaiOS, a feature phone platform built on the ashes of Firefox OS, adds Facebook, Twitter and Google apps" (https://techcrunch.com/2018/02/26/kaios-a-feature-phone-platform-built-on-the-ashes-of-firefox-os-adds-facebook-twitter-and-google-apps/). *TechCrunch*. 26 February 2018. Retrieved 2021-03-17.
14. Russell, Alex. "Progressive Web Apps: Escaping Tabs Without Losing Our Soul" (https://infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul/). Retrieved June 15, 2015.
15. Evans, Jonny (26 January 2018). "Apple goes back to the future with web apps" (https://www.computerworld.com/article/3251173/apple-goes-back-to-the-future-with-web-apps.html). *Computerworld*. Retrieved 23 May 2019.
16. Ladage, Aaron (17 April 2018). "Progressive Web Apps Are Here and They're Changing Everything" (https://www.degdigital.com/insights/progressive-web-apps/). *DEG*. Retrieved 23 May 2019.
17. "Can I use... Support tables for HTML5, CSS3, etc" (https://caniuse.com/serviceworkers). *caniuse.com*. Retrieved 2021-05-16.
18. "Progressive Web Apps on Windows overview" (https://docs.microsoft.com/en-us/microsoft-edge/progressive-web-apps-chromium/). *Microsoft Edge Documentation*. 13 March 2021. Retrieved 13 March 2021.
19. LePage, Pete (4 June 2019). "Progressive Web Apps on Desktop" (https://developers.google.com/web/progressive-web-apps/desktop). *Google Developers*. Retrieved 13 September 2019.
20. "eBay PWA" (https://bugzilla.mozilla.org/show_bug.cgi?id=1933032). Mozilla. Retrieved 25 December 2024.
21. "Get your PWA on" (https://vivaldi.com/blog/vivaldi-gets-more-private-delivers-an-all-new-capture-pwa-support/). *Vivaldi Browser*. 2021-10-07. Retrieved 2021-10-11.
22. Angle, Patrick; Avenard, Jean-Yves; Caceres, Marcos; Cannon, Ada Rose; Carlson, Eric; Davidson, Garrett; Davis, Jon; Dubost, Karl; Eidson, Brady (2023-06-06). "News from WWDC23: WebKit Features in Safari 17 beta" (https://webkit.org/blog/14205/news-from-wwdc23-webkit-features-in-safari-17-beta/). *WebKit*. Retrieved 2023-06-14.
23. Angle, Patrick; Caceres, Marcos; Caliman, Razvan; Davis, Jon; Eidson, Brady; Hatcher, Timothy; Niwa, Ryosuke; Simmons, Jen (2023-03-27). "WebKit Features in Safari 16.4" (https://webkit.org/blog/13966/webkit-features-in-safari-16-4/). *WebKit*. Retrieved 2023-06-14.
24. "The App Shell Model" (https://developers.google.com/web/fundamentals/architecture/app-shell).

25. Osmani, Addi. "The App Shell Model | Web Fundamentals" (https://developers.google.com/web/fundamentals/architecture/app-shell). *Google Developers*. Retrieved 23 May 2019.
26. Russell, Alex. "What, Exactly, Makes a Progressive Web App" (https://infrequently.org/2016/09/what-exactly-makes-something-a-progressive-web-app/). Retrieved October 18, 2016.
27. "What does it take to be installable?" (https://web.dev/install-criteria/). *web.dev*. Retrieved 2021-05-19.
28. "Progressive Web App" (https://developers.google.com/web/progressive-web-apps). *web.dev*. Retrieved June 15, 2015.
29. "Service worker caching and HTTP caching" (https://web.dev/service-worker-caching-and-http-caching/). *web.dev*. Retrieved 2021-05-19.
30. W3C "Web App Manifest", Working Draft (https://www.w3.org/TR/appmanifest/), retrieved 12 September 2016.
31. "Discoverable" (https://developer.mozilla.org/en-US/Apps/Progressive/Discoverable). *Mozilla Developer Network*. Retrieved 2017-04-24.
32. "Network independent" (https://developer.mozilla.org/en-US/Apps/Progressive/Network_independent). *Mozilla Developer Network*. Retrieved 2017-04-24.
33. "Instant Loading Web Apps with an Application Shell Architecture" (https://developers.google.com/web/updates/2015/11/app-shell). *Google Developers*. Retrieved 2017-04-24.
34. "Web Manifest Docs on MDN" (https://developer.mozilla.org/en-US/docs/Web/Manifest). *MDN Web Docs*.
35. "What's new on iOS 12.2 for Progressive Web Apps" (https://medium.com/@firt/whats-new-on-ios-12-2-for-progressive-web-apps-75c348f8e945). *Medium*. 27 March 2019.
36. "Configuring Web Applications" (https://developer.apple.com/library/archive/documentation/AppleApplications/Reference/SafariWebContent/ConfiguringWebApplications/ConfiguringWebApplications.html). *Safari Web Content Guide*.
37. "Introduction to Service Worker | Web" (https://developers.google.com/web/ilt/pwa/introduction-to-service-worker). *Google Developers*. 1 May 2019. Retrieved 23 May 2019.
38. "WebAssembly Concepts" (https://developer.mozilla.org/en-US/docs/WebAssembly/Concepts). *MDN*. Retrieved 14 August 2018.
39. World Wide Web Consortium. "WebAssembly Core Specification" (https://www.w3.org/TR/wasm-core-1/). *World Wide Web Consortium (W3)*. Retrieved 2024-05-06.
40. "WebAssembly 1.0 Becomes a W3C Recommendation and the Fourth Language to Run Natively in Browsers" (https://www.infoq.com/news/2019/12/webassembly-w3c-recommendation/). *InfoQ*. Retrieved 2024-05-06.
41. "Programming Languages Software Award" (https://www.sigplan.org/Awards/Software/). *www.sigplan.org*. Retrieved 2024-05-06.
42. "Web Storage API" (https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API). *MDN*. Retrieved 14 August 2018.
43. "Concepts behind IndexedDB" (https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API/Basic_Concepts_Behind_IndexedDB). *MDN*. Retrieved 14 August 2018.
44. Shankland, Stephen (30 July 2020). "Twitter's app is helping stop phones from strangling the web" (https://www.cnet.com/tech/mobile/twitter-shows-why-web-apps-can-work-on-your-android-phone-or-iphone/). *CNET*. Retrieved 11 February 2023.
45. "12 Best Examples of Progressive Web Apps (PWAs) in 2021" (https://www.simicart.com/blog/progressive-web-apps-examples/). *SimiCart*. 2021-02-22. Retrieved 2021-05-16.
46. Osmani, Addy (30 November 2017). "A Pinterest Progressive Web App Performance Case Study" (https://medium.com/dev-channel/a-pinterest-progressive-web-app-performance-case-study-3bd6ed2e6154). *ChromiumDev team*. Retrieved 10 February 2023.

47. Gazdecki, Andrew (9 March 2018). "Why Progressive Web Apps Will Replace Native Mobile Apps" (https://www.forbes.com/sites/forbestechcouncil/2018/03/09/why-progressive-web-apps-will-replace-native-mobile-apps/?sh=154c9d2a2112). *Forbes*. Retrieved 10 February 2023.
48. MSEdgeTeam. "Publish your Progressive Web App to the Microsoft Store - Microsoft Edge Development" (https://docs.microsoft.com/en-us/microsoft-edge/progressive-web-apps-chromium/microsoft-store). *docs.microsoft.com*. Retrieved 2021-05-16.
49. "The first batch of Windows 10 Progressive Web Apps is here" (https://www.windowscentral.com/first-batch-windows-10-progressive-web-apps-here). *Windows Central*. 2018-04-07. Retrieved 2021-05-16.

External links

- Web Applications Working Group (https://www.w3.org/2019/webapps/) index of standards