


How to make PWAs installable


Jump to:

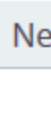

[Requirements](#)

[Add to home screen](#)

[Summary](#)

 Previous

 Overview: Progressive

 Next 

In the last article, we read about how `js13kPWA` works offline thanks to its [service worker](#), but we can go even further and allow users to install the web app on supporting mobile browsers, just as if it was a native app. This article explains how to achieve this, using the web manifest and a feature called add to home screen.

These technologies allow the app to be launched directly from the device's home screen instead of starting the browser manually and typing in the URL. Your web app can sit next to native applications as first class citizens. It is therefore easier to access, and you can also specify that an app should run fullscreen without the surrounding browser chrome, making it feel even more like a native app.

Requirements

To make the website installable, it needs the following things in place:

Jump to:

[Requirements](#)

[Add to home screen](#)

[Summary](#)

- The website to be served from a secure (HTTPS) domain
- An icon to represent the app on the device
- A service worker registered, to make the app work offline (this is required only by Chrome for Android currently)

The manifest file

The key element is a web manifest file, which lists all the information about the website in a JSON format.

It usually resides in the root folder of a web app. It contains useful information, such as the app's title, paths to different-sized icons that can be used to represent the app on a mobile OS (for example, as the home screen icon), and a background color to use in loading or splash screens. This information is needed for the browser to present the web app properly when installing, and on the home screen.

The `js13kpa.webmanifest` file of the `js13kPWA` web app is included in the `<head>` section of the `index.html` file via the following line of code:

```
1 | <link rel="manifest" href="js13kpa.webmanifest">
```

 **Note:** There are a few common extensions that have been used for manifests in the past: `manifest.webapp` was popular in Firefox OS app manifests, and many use `manifest.json` for web manifests as the contents are organized in a JSON structure. However, the `.webmanifest` extension is explicitly mentioned in the [W3C manifest specification](#), so let's stick to that.

The content of the file looks like this:

```
1 | {
2 |   "name": "js13kGames Progressive Web App",
3 |
4 |   "short_name": "js13kGames",
5 |
6 |   "icons": [
7 |     {
8 |       "src": "icons/icon-32.png",
9 |       "sizes": "32x32",
10 |      "type": "image/png"
11 |     },
12 |     // ...
13 |     {
14 |       "src": "icons/icon-512.png",
15 |       "sizes": "512x512",
16 |       "type": "image/png"
17 |     }
18 |   ],
19 |   "start_url": "/pwa-examples/js13kpa/index.html",
20 |   "display": "fullscreen",
21 |   "theme_color": "#B12A34",
22 |   "background_color": "#B12A34"
23 | }
```

Most of the fields are self-explanatory, but let's break down the document and explain them in detail:

- `name`: The full name of your web app.
- `short_name`: Short name to be shown on the home screen.
- `description`: A sentence or two explaining what your app does.
- `icons`: A bunch of icon information — source URLs, sizes, and types. Be sure to include at least a few, so that one that fits best will be chosen for the user's device.
- `start_url`: The index document to launch when starting the app.
- `display`: How the app is displayed; can be fullscreen, standalone, minimal-ui, or browser
- `theme_color`: A primary color for the UI, used by operating system
- `background_color`: A color for background, used during install and on the splash

Jump to:

[Requirements](#)

[Add to home screen](#)

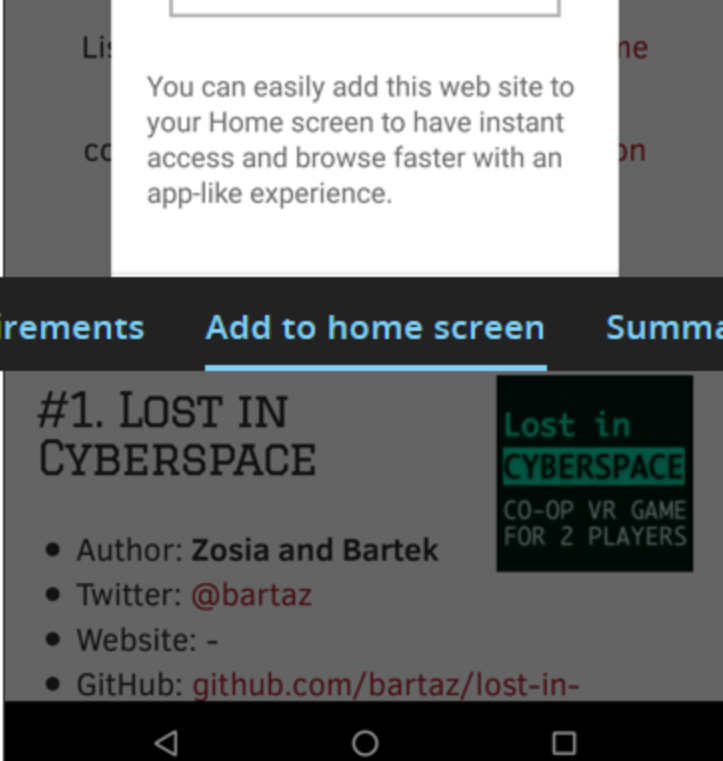
[Summary](#)

The bare minimum requirement for a web manifest is `name` and at least one icon (with `src`, `size` and `type`). The `description`, `short_name`, and `start_url` are recommended. There are even more fields you can use than listed above — be sure to check the [Web App Manifest reference](#) for details.

Add to home screen

"Add to home screen" (or `a2hs` for short) is a feature implemented by mobile browsers that takes the information found in an app's web manifest and uses them to represent the app on the device's home screen with an icon and name. This only works if the app meets all the necessary requirements, as described above.

When the user visits the PWA with a supporting mobile browser, it should display a banner indicating that it's possible to install the app as a PWA.

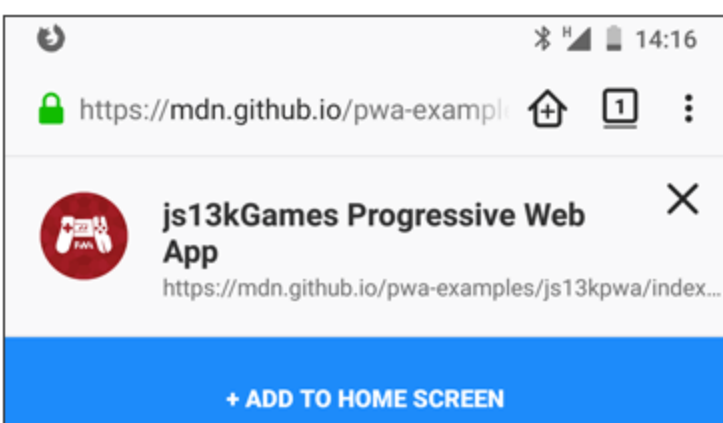


Jump to:

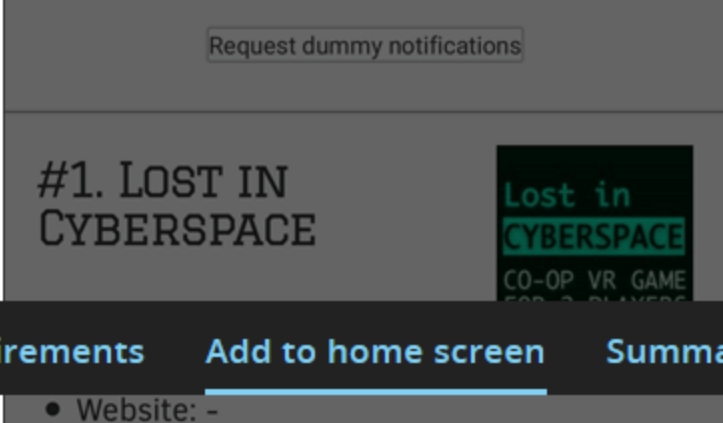
[Requirements](#)

[Add to home screen](#)

[Summary](#)



After the user clicks this banner, the install banner is shown. That banner is automatically created by the browser, based on the information from the manifest file — the name and icon are visible on the prompt.

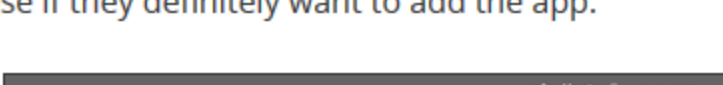


Jump to:

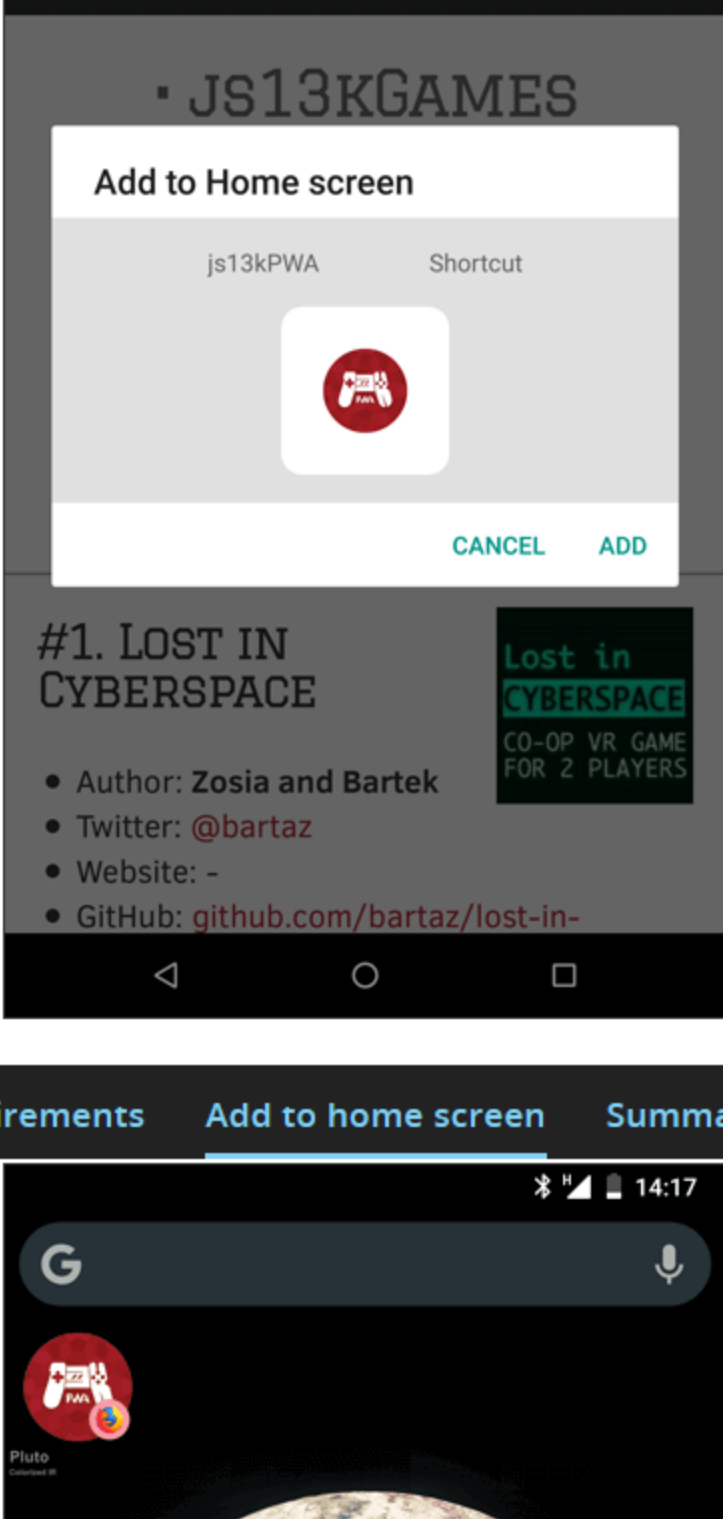
[Requirements](#)

[Add to home screen](#)

[Summary](#)



If the user clicks the button, there is a final step showing what the app will look like, and letting the user choose if they definitely want to add the app.



Jump to:

[Requirements](#)

[Add to home screen](#)

[Summary](#)



After that, the user can launch it and start using it immediately. Notice that PWAs sometimes (depending on the browser or mobile operating system you are using) have a small browser image on the bottom right corner of their icon to inform the user about their web nature.

Splash screen

Jump to:

[Requirements](#)

[Add to home screen](#)

[Summary](#)



The icon and the theme and background colors are used to create this screen.

Summary

Jump to:

[Requirements](#)

[Add to home screen](#)

[Summary](#)


In this article, we learned about how we can make PWAs installable, using web manifest and Add to home screen.

For more information on Add to Home screen, be sure to read our [Add to Home screen guide](#). Browser support is currently limited to Firefox for Android 58+, Mobile Chrome and Android Webview 31+, and Opera for Android 32+, but this should improve in the near future.

Now let's move to the last piece of the PWA puzzle — re-engagement via push notifications.

 Previous

 Overview: Progressive

 Next 


Jump to:

[Requirements](#)

[Add to home screen](#)

[Summary](#)

 Contributors to this page: [chrisdavidmills](#), [end3r](#), [jswisher](#)

 Last updated by: [chrisdavidmills](#), May 1, 2018, 4:14:16 AM

Learn the best of web development

Get the latest and greatest from MDN delivered straight to your inbox.

