The Web App Manifest

By Matt Gaunt

Matt is a contributor to Web**Fundamentals**

By Paul Kinlan

Paul is a Developer Advocate

The web app manifest is a simple JSON file that tells the browser about your web application and how it should behave when 'installed' on the users mobile device or desktop. Having a manifest is required by Chrome to show the Add to Home Screen prompt.

A typical manifest file includes information about the app `name`, `icons` it should use, the `start_url` it should start at when launched, and more.

Create the manifest

A complete `manifest.json` file for a progressive web app.

```
{
  "short_name": "Maps",
  "name": "Google Maps",
  "icons": [
    {
      "src": "/images/icons-192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "/images/icons-512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": "/maps/?source=pwa",
  "background_color": "#3367D6",
  "display": "standalone",
  "scope": "/maps/",
  "theme_color": "#3367D6"
}
```

Note: See the add to home screen criteria for the specific properties that are required to show the add to home screen prompt.

Tell the browser about your manifest

When you have created the manifest add a `link` tag to all the pages that encompass your web app:

```
<link rel="manifest" href="/manifest.json">
```

Key manifest properties

`short_name` and/or `name`

You must provide at least the **`short_name`** or **`name`** property. If both are provided, **`short_name`** is used on the user's home screen, launcher, or other places where space may be limited. **`name`** is used on the app install prompt.

```
"short_name": "Maps",
"name": "Google Maps"
```

`icons`

When a user adds your site to their home screen, you can define a set of icons for the browser to use. These icons are used in places like the home screen, app launcher, task switcher, splash screen, etc.

`icons` is an array of image objects, each object should include the **`src`**, a **`sizes`** property, and the **`type`** of image.

```
"icons": [
  {
    "src": "/images/icons-192.png",
    "type": "image/png",
    "sizes": "192x192"
  },
  {
    "src": "/images/icons-512.png",
    "type": "image/png",
    "sizes": "512x512"
```

```
  }
]
```

Success: include a 192x192 pixel icon and a 512x512 pixel icon. Chrome will automatically scale the icon for the device. If you'd prefer to scale your own icons and adjust them for pixel-perfection, provide icons in increments of 48dp.

start_url

The `start_url` tells the browser where your application should start when it is launched, and prevents the app from starting on whatever page the user was on when they added your app to their home screen.

Your `start_url` should direct the user straight into your app, rather than a product landing page. Think about the what the user will want to do once they open your app, and place them there.

```
"start_url": "/?utm_source=a2hs"
```

Success: add a query string to the end of the `start_url` to track how often your app is launched.

background_color

The `background_color` property is used on the splash screen when the application is first launched.

display

You can customize what browser UI is shown when your app is launched. For example, you can hide the address bar and browser chrome. Or games may want to go completely full screen.

```
"display": "standalone"
```

Parameters

value	Description
`fullscreen`	Opens the web application without any browser UI and takes up the

	entirety of the available display area.
standalone	Opens the web app to look and feel like a standalone native app. The app runs in it's own window, separate from the browser, and hides standard browser UI elements like the URL bar, etc.
minimal-ui	**Not supported by Chrome** This mode is similar to **fullscreen**, but provides the user with some means to access a minimal set of UI elements for controlling navigation (i.e., back, forward, reload, etc).
browser	A standard browser experience.

Success: In order to show the Add to Home Screen Prompt, **display** must be set to standalone.

orientation

You can enforce a specific orientation, which is advantageous for apps that work in only one orientation, such as games. Use this selectively. Users prefer selecting the orientation.

```
"orientation": "landscape"
```

scope

The **scope** defines the set of URLs that the browser considers within your app, and is used to decide when you've left your app, and should be bounced back out to a browser tab. The **scope** controls the url structure that encompasses all the entry and exit points in your web app. Your **start_url** must reside within the **scope**.

```
"scope": "/maps/"
```

A few other tips:

- If you don't include a **scope** in your manifest, then the default implied **scope** is the directory that your web app manifest is served from.
- The **scope** attribute can be a relative path (**../**), or any higher level path (**/**) which would allow for an increase in coverage of navigations in your web app.
- The **start_url** must be in the scope.
- The **start_url** is relative to the path defined in the **scope** attribute.
- A **start_url** starting with **/** will always be the root of the origin.

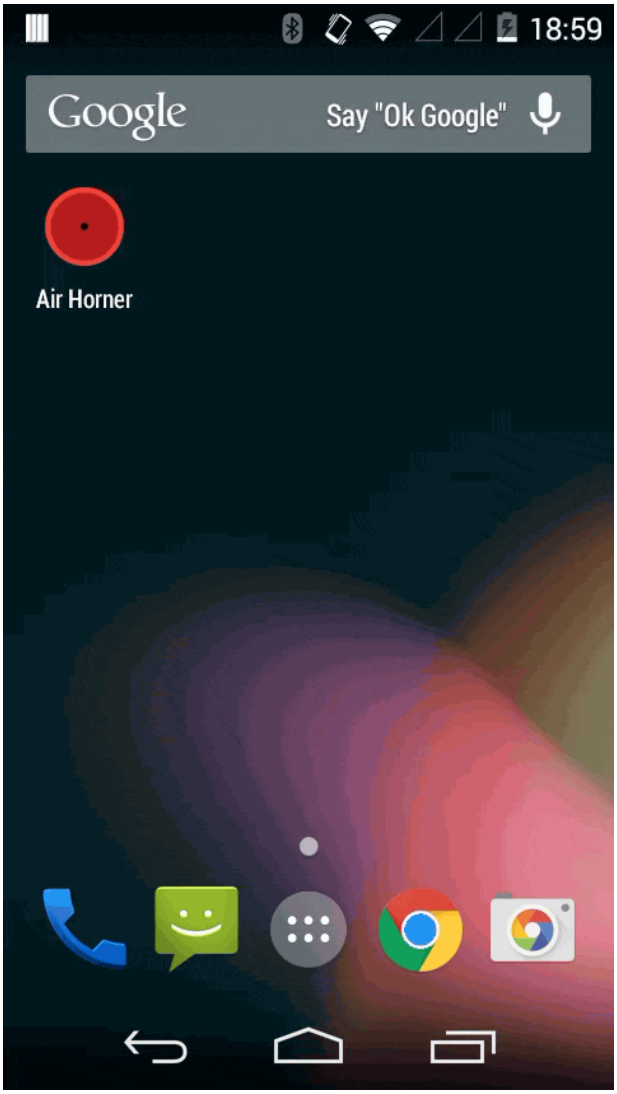
`theme_color`

The `theme_color` sets the color of the tool bar, and in the task switcher.

```
"theme_color": "#3367D6"
```

Success: the `theme_color` should match the `meta` theme color specified in your document head.

Splash screens



Background color for launch screen

When your app first launches, it can take a moment for the browser to spin up, and the initial content to begin rendering. Instead of showing a white screen that may look to the user like the app is stall, Chrome will show a splash screen, until the first paint.

Chrome will automatically create the splash screen from the manifest properties, including:

- `name`
- `background_color`
- `icons`

The `background_color` should be the same color as the load page, to provide a smooth transition from the splash screen to your app.

Icons used for the splash screen

Chrome will choose the icon that closely matches the 128dp icon for that device. 128dp is the ideal size for the image on the splash screen, and means no scaling will be applied to the image.

Again, providing a 192px and a 512px icon will be sufficient for most cases, but you can provide additional icons as necessary.

Feedback

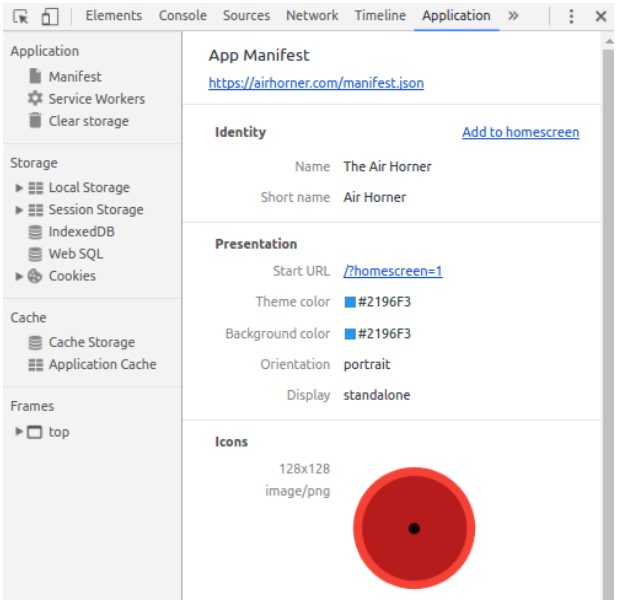
Was this page helpful?

YES NO

Great! Thank you for the feedback.

Sorry to hear that. Please open an issue and tell us how we can improve.

Test your manifest



Manifest tab of Chrome DevTools

To verify your manifest is setup correctly, you can use the **Manifest** tab in the **Application** panel of Chrome DevTools.

If you want to manually verify that your web app manifest is set up correctly, use the **Manifest** tab on the **Application** panel of Chrome DevTools.

This tab provides a human-readable version of many of your manifest's properties. You can also simulate Add to Home Screen events from here. See Testing the app install banner for more on this topic.

If you want an automated approach towards validating your web app manifest, check out Lighthouse. Lighthouse is a web app auditing tool that you run as a Chrome Extension or as an NPM module. You provide Lighthouse with a URL, it runs a suite of audits against that page, and then displays the results in a report.

What's next?

- If you're using a web app manifest, you'll probably want set up an app install banner as well.
- A complete reference to the web app manifest is available on the Mozilla Developer Network.
- If you want feature descriptions from the engineers who created web app manifests, you can read the W3C Web App Manifest Spec.

Last updated July 24, 2018.