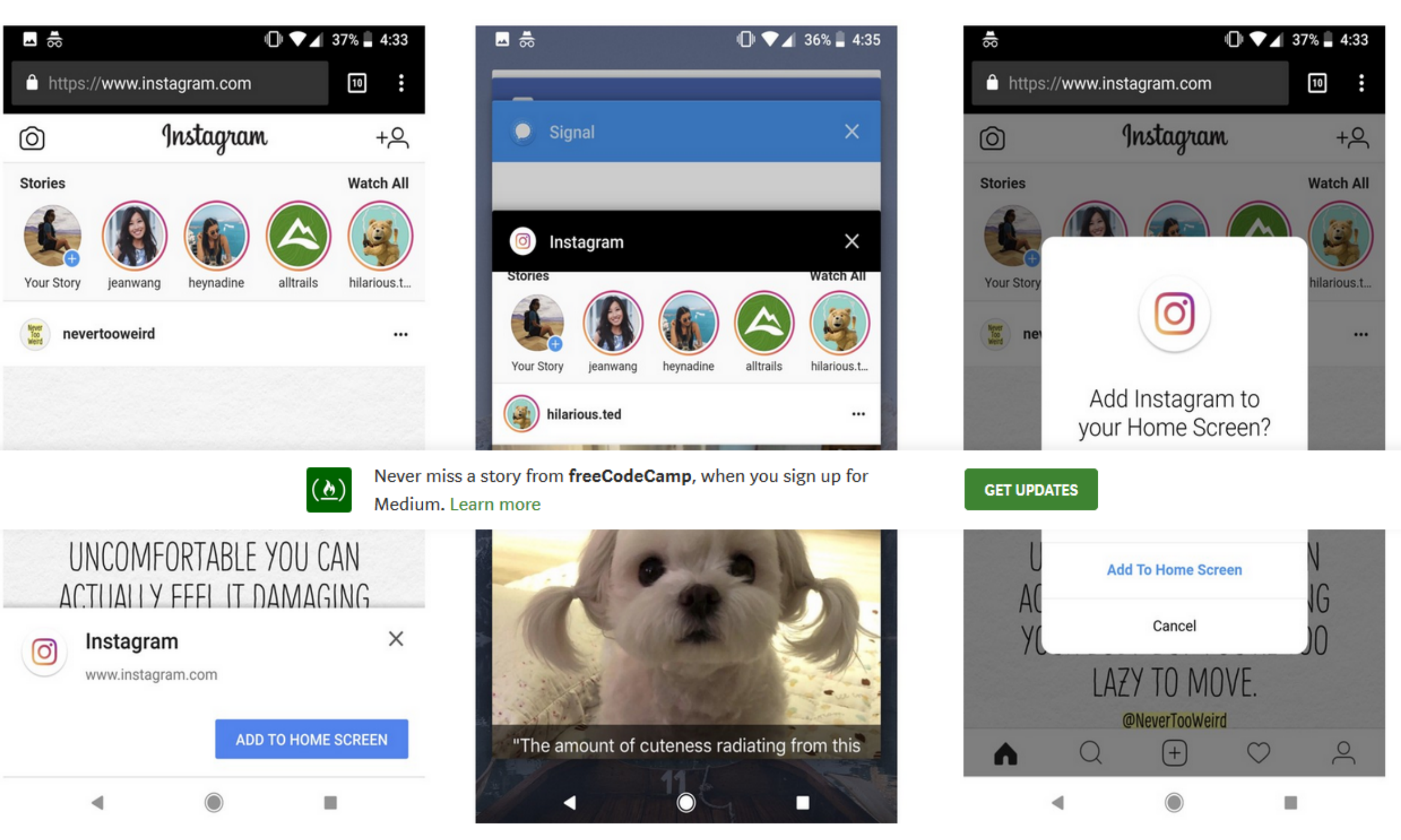
Shruti Kapoor Follow
Software Engineer @ PayPal. I write about JavaScript. Tweet me @shrutikapoor08
Jul 20 · 5 min read

Progressive Web Apps 101: the What, Why and How

What is a Progressive Web App? Why do we need one? How can we build one?



Instagram's Progressive Web Application

Have you ever seen an "Add to Home Screen" banner, like above, while browsing a website? When you click the button, the "application" installs itself in the background. When you open this application that now sits in your app drawer, you can browse the same experience you were doing on your browser, but now right in your mobile phone.

What you have now is a mobile app that was downloaded from a web application. All this, without even have to see the face of an app store.

Getting the app was so easy! But that's not even the best part. When you open this app, you will be able to browse the content even when you do not have internet. You have offline access to the app! How cool is that?

What you've come across is a Progressive Web App (PWA). A PWA lets you install the application from the browser window itself, is available on your phone like a native app, and works offline, just like a native app.

But what does it really mean for a web app to be progressive? Let's take a deeper dive into what progressive web apps are, why I think they're better than native apps, and what makes them different from traditional web apps.

What is a Progressive Web App (PWA)?

The term Progressive Web App was coined by Alex Russell and Frances Berriman. In Alex's words:

> *Progressive Web Apps are just websites that took all the right vitamins.*

Top highlight

It isn't a new framework or technology. It is a set of best practices to make a web application function similar to a desktop or mobile application. The dream is to have an experience so uniform and seamless that the user is unable to tell the difference between a Progressive Web App and a native mobile app.

Progressive web applications deliver user experiences through progressive enhancement. It essentially means that a PWA will perform the same functions on a new iPhone 8 as it would on an older generation iPhone. Sure, some features may not be available, but the app continues to work and perform like it should.

Why do we need a Progressive Web App?

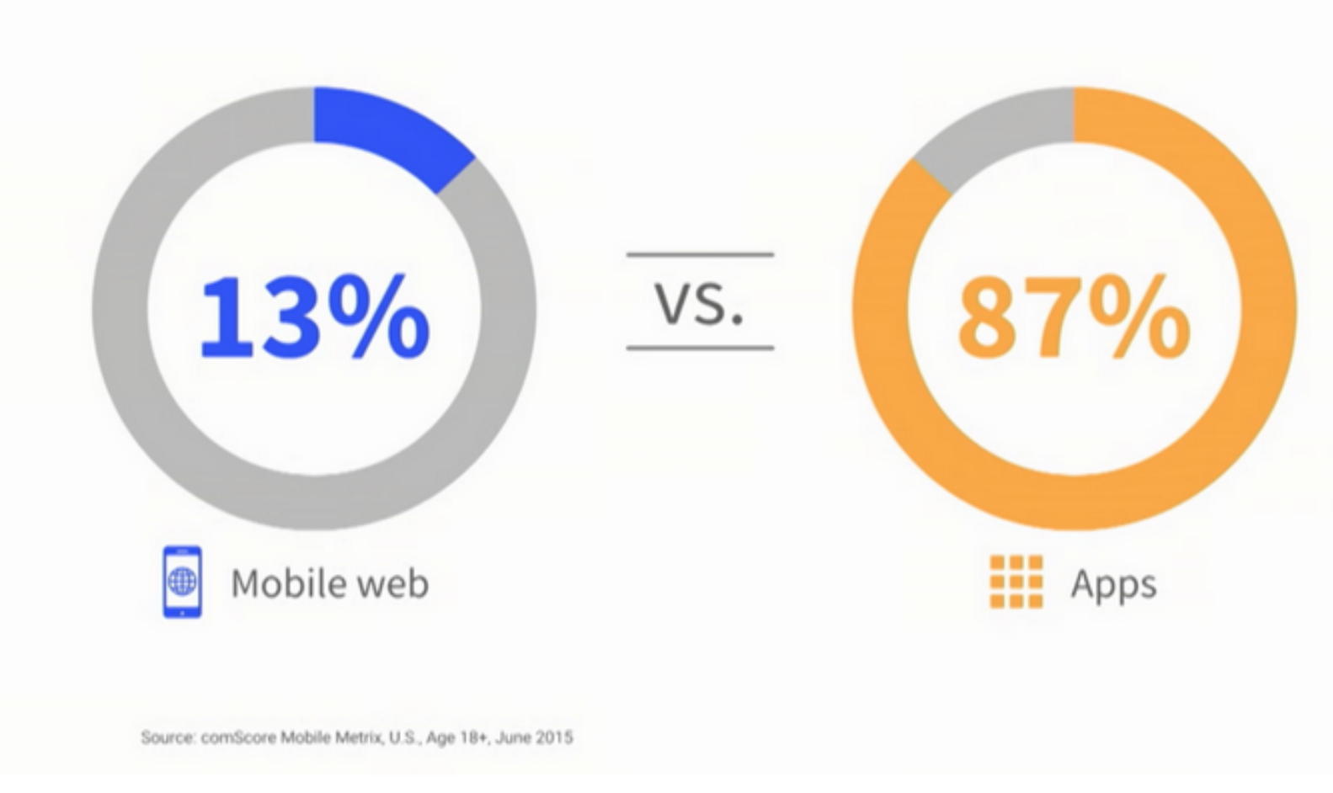
Before we understand why we need a progressive web app, let's talk about some of the challenges we are facing today with native and web apps.

Internet speed: you may not realize this depending on where you live, but 60% of the world's population is still using 2G internet. Even in the US, some people have to use dialup to access internet.

Slow website load: Do you know how long a user waits to click the "Close X" button if a website is too slow? Three seconds! 53% of users abandon a website if it is too slow.

High friction: People don't want to install native apps. An average user installs 0 applications in a month.

User engagement: Users spend most of their time in native apps, but mobile web reach is almost three times that of native apps. Hence, most of the users are not actively engaged. However, users are spending 80% of their time on only their top three native apps.



Source: comScore Mobile Metrix, U.S., Age 18+, June 2015

User engagement on mobile web vs apps

PWAs help solve these problems. There are multiple reasons for using a progressive web app, but here are some of the top capabilities it provides:

1. Fast: PWAs provide experiences that are consistently fast. From the moment a user downloads an app to the moment they start interacting with it, everything happens really fast. Because you can cache the data, it is extremely fast to start the app again even without hitting the network.
2. Integrated user experience: PWAs feel and behave like native apps. They sit in a user's home screen, send push notifications like native apps, and have access to a device's functionalities like native apps. The experience feels seamless and integrated.
3. Reliable experience: With the help of service workers, we can reliably paint a picture on a user's screen even when network has failed.
4. Engaging: Because we can send notifications to a user, we can really drive the engagement up by keeping the user notified and engaged with the app.

In short, it is **FIRE.**

How to build a Progressive Web App

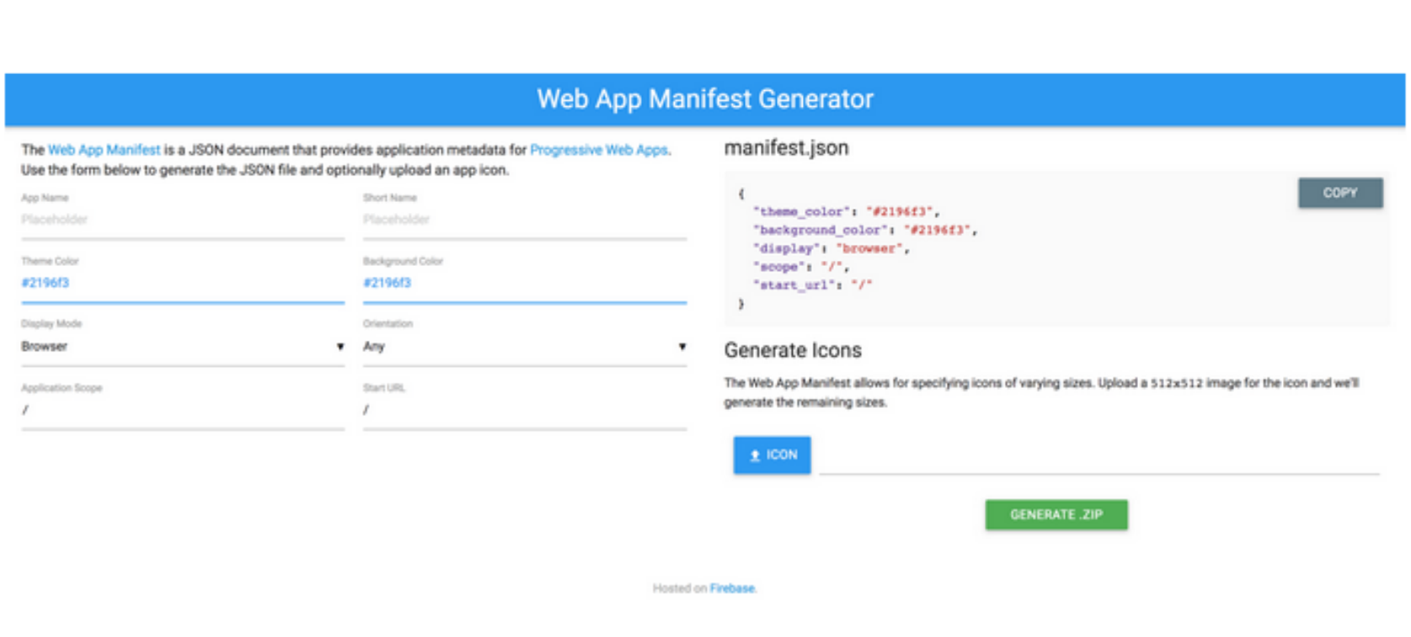
Google has published a checklist of items for Progressive Web apps. I will go over four minimum requirements for an application to be a PWA:

1. Web App Manifest

```
{
  "name": "Trending Meme",
  "short_name": "Meme",
  "theme_color": "#0f1113",
  "background_color": "#111316",
  "display": "standalone",
  "orientation": "portrait",
  "Scope": "/",
  "start_url": "/",
  "icons": [
    {
      "src": "images/icons/icon-72x72.png",
      "sizes": "72x72",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-96x96.png",
      "sizes": "96x96",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-128x128.png",
      "sizes": "128x128",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-144x144.png",
      "sizes": "144x144",
      "type": "image/png"
```

A sample manifest.json file

This is just a `json` file that gives meta information about the web app. It has information like the icon of the app (which a user sees after installing it in their app drawer), background color of the app, name of the app, short name, and so on. We can write this manifest file ourselves or we can use tools to generate one for us.



The Web App Manifest is a JSON document that provides application metadata for Progressive Web Apps. Use the form below to generate the JSON file and optionally upload an app icon.

Theme Color: #2196f3
Background Color: #2196f3
Display Mode: Browser
Orientation: Any
Application Scope: /
Start URL: /

```
{
  "theme_color": "#2196f3",
  "background_color": "#2196f3",
  "display": "browser",
  "scope": "/",
  "start_url": "/"
}
```

The Web App Manifest allows for specifying icons of varying sizes. Upload a 512x512 image for the icon and we'll generate the remaining sizes.

You can auto-generate manifest file using Google's tools.

2. Service Workers

Service Workers are event-driven workers that run in the background of an application and act as a proxy between the network and application. They are able to intercept network requests and cache information for us in the background. This can be used to load data for offline use. They are a `javascript` script that listens to events like fetch and install, and they perform tasks.

Here is a sample `serviceworker.js`

```
self.addEventListener('fetch', event => {
    //caching for offline viewing
    const {request} = event;
    const url = new URL(request.url);
    if(url.origin === location.origin) {
        event.respondWith(cacheData(request));
    } else {
        event.respondWith(networkFirst(request));
    }

});

async function cacheData(request) {
    const cachedResponse = await caches.match(request);
    return cachedResponse || fetch(request);
}
```

3. Icon

This is used to provide an app icon when a user installs the PWA in their application drawer. A jpeg image will just be fine. The manifest tool I highlighted above helps in generating icons for multiple formats, and I found it very useful.

4. Served over HTTPS

In order to be a PWA, the web application must be served over a secure network. With services like Cloudfare and LetsEncrypt, it is really easy to get an SSL certificate. Being a secure site is not only a best practice, it also establishes your web application as a trusted site for users demonstrating trust and reliability, and avoiding middle man attacks.

Note: This is part 1 of 2 part series. In the next part, we will create a Progressive Web App from scratch with a skeleton index.html. Stay tuned!

Did you learn something new? Have comments? Know a DevJoke? Tweet me @shrutikapoor08

Shruti Kapoor
@shrutikapoor08

Why do programmers get confused between Halloween and Christmas?
Because OCT 31 = DEC 25 #DevJokes #WorkChat

10:23 PM - Jun 22, 2018

26 See Shruti Kapoor's other Tweets

Web Development React Progressive Web App JavaScript Tech

Like what you read? Give Shruti Kapoor a round of applause.

From a quick cheer to a standing ovation, clap to show how much you enjoyed this story.

5.8K 16

Shruti Kapoor Follow
Software Engineer @ PayPal. I write about JavaScript. Tweet me @shrutikapoor08

freeCodeCamp Follow
Our community publishes stories worth reading on development, design, and data science.

from freeCodeCamp
to get better at writing CSS
Thomas Lombart
10 min read 10.1K

More from freeCodeCamp
How to scale Microservices with Message Queues, Spring Boot, and
Daniele Polencic
16 min read 1.1K

More from freeCodeCamp
Docker ENTRYPOINT & CMD: Dockerfile best practices
ryanwhocodes
5 min read 551

Responses

Write a response...

Conversation between Philippe Bernard and Shruti Kapoor.

Philippe Bernard
Jul 21

Generating a web app manifest and icons? Why not using RealFaviconGenerator? The fact that I'm the author of this service does not make me biased at all 😅

15 1 response

Shruti Kapoor
Jul 23

haha good to know!

Conversation between Mukesh Gupta and Shruti Kapoor.

Mukesh Gupta
Jul 22

Great post!

Eagerly waiting for the part 2! 😊😊

32 1 response

Shruti Kapoor
Jul 23

Thank you! Coming up soon :)

59

Applause from Shruti Kapoor (author)

Garth Mountain
Jul 22

…apologies, just seen the publish date on this one, so part 2 is obviously still under construction.

3

Conversation with Shruti Kapoor.

Ana Mengote Baluca
Jul 21

Thank you for this! I'm new in the development game, and this seems like a must-have for an idea that's been simmering on my mind, with a market in a country where internet connection is slow and expensive.

3 1 response

Shruti Kapoor
Jul 23

Absolutely a great idea in a market like that.

10

Conversation with Shruti Kapoor.

Stefan Walkner
Jul 21

but stuff like Push Notifications and In App Purchases are not possible, right?

1 response

Shruti Kapoor
Jul 23

Push notifications is definitely available. I haven't worked with in app purchases myself but you should check out Eric Elliott's article

Show all responses