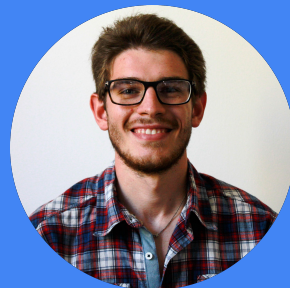




PWA from scratch



Aldo D'Aquino

 /daquinoaldo

 <http://ald.ooo>





from scratch

Wi-Fi
Password

GATE WIFI - GUEST
Gate2018!

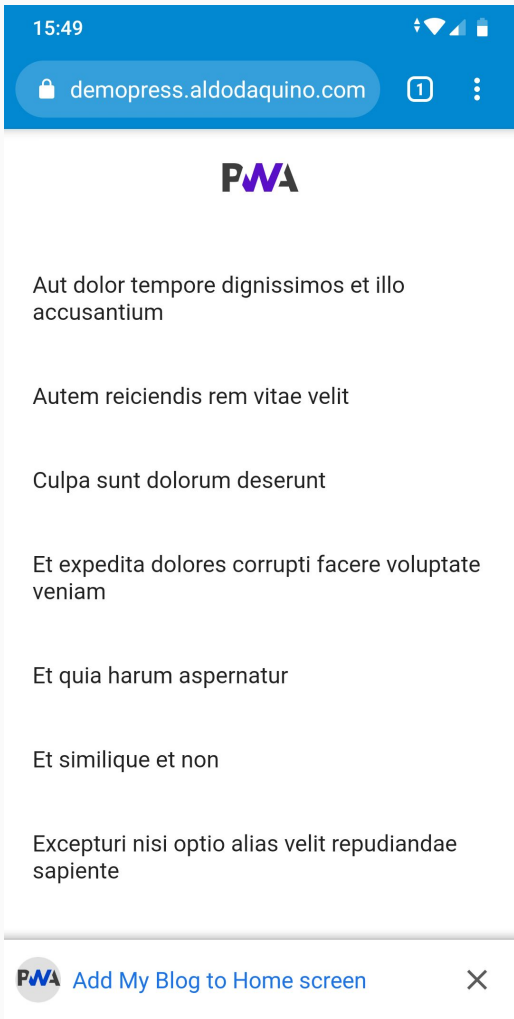


Aldo D'Aquino

 /daquinoaldo

 ald.ooo





Web

URL

Cross
platform

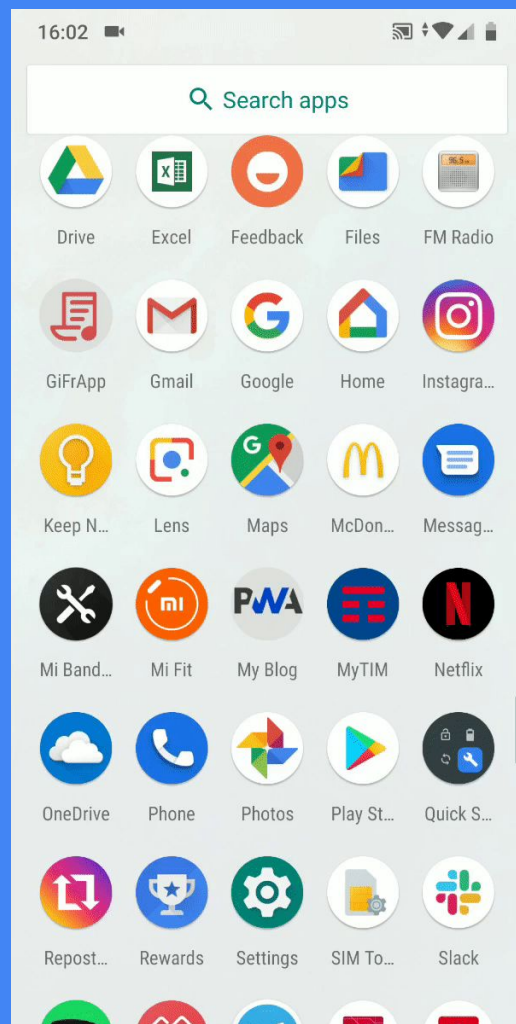
No need for
install

App

Handy

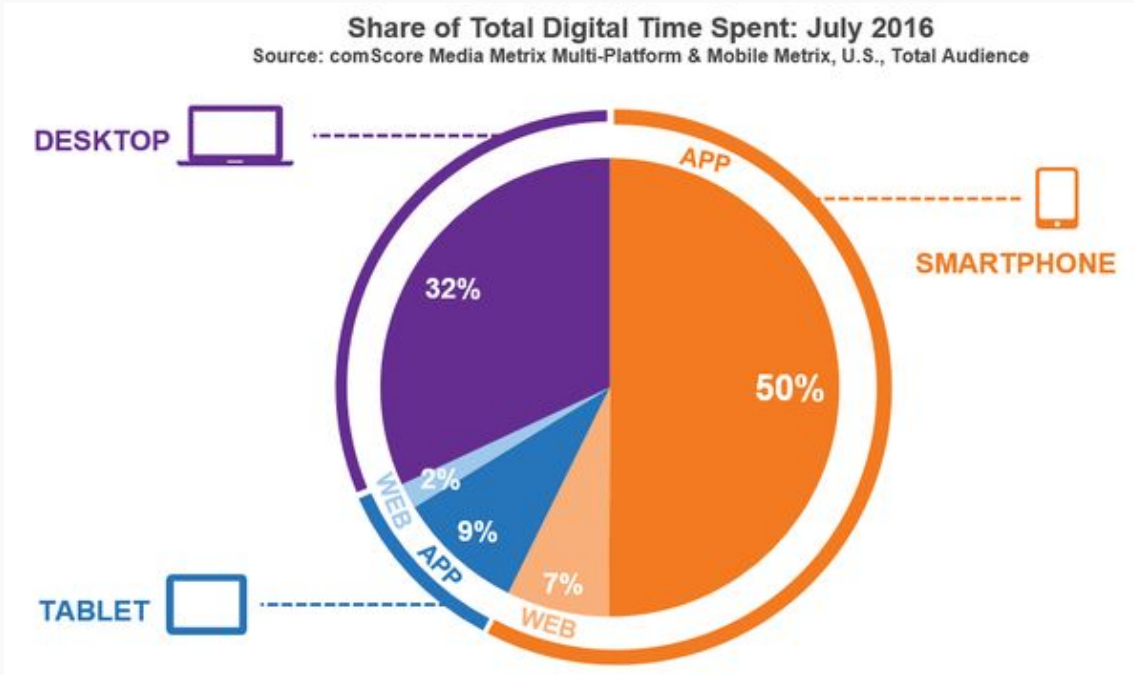
Natural UX
on mobile

Fast &
Offline



Web + App

Gain twice!



Key aspects

Responsive

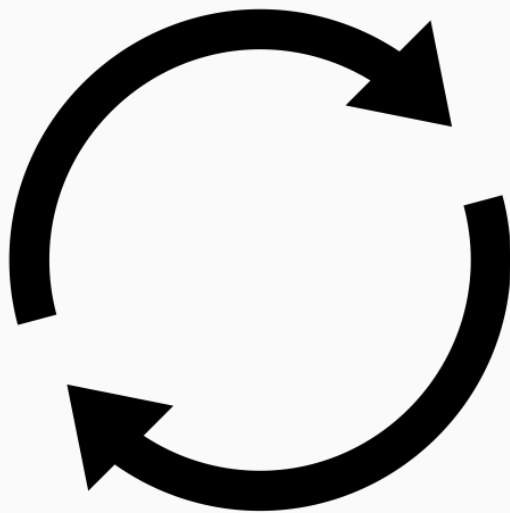


Secure



https://

Always fresh



Connectivity independent

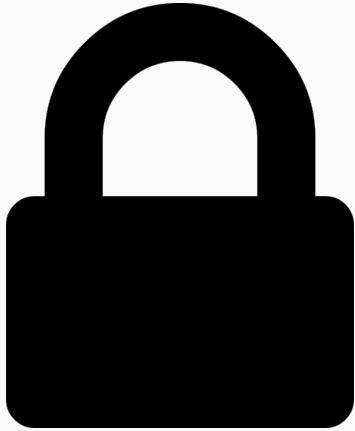


Immersive

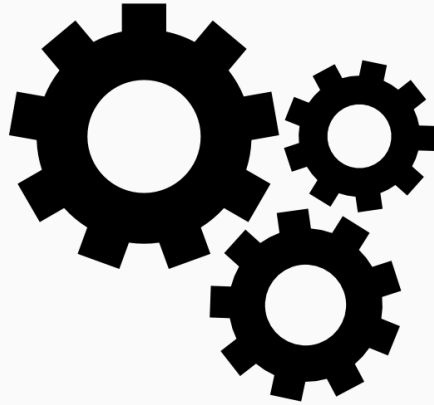


What we need

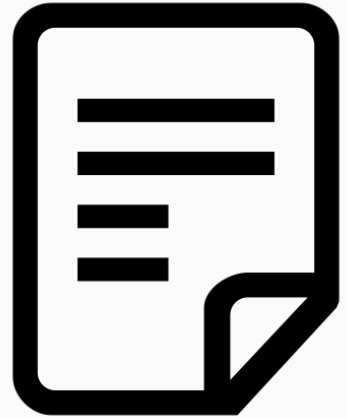
What we need



HTTPS

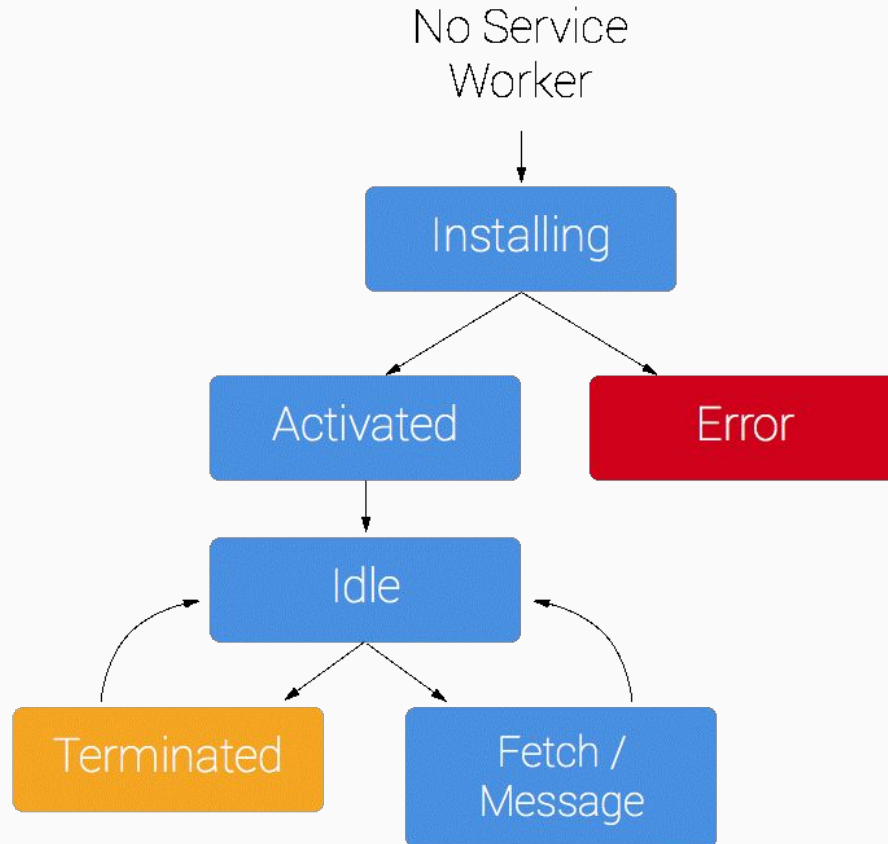


ServiceWorker.js



Manifest.json

Service Worker life cycle



UX first

How to design a great UI



How to design a great UI

1. Be app

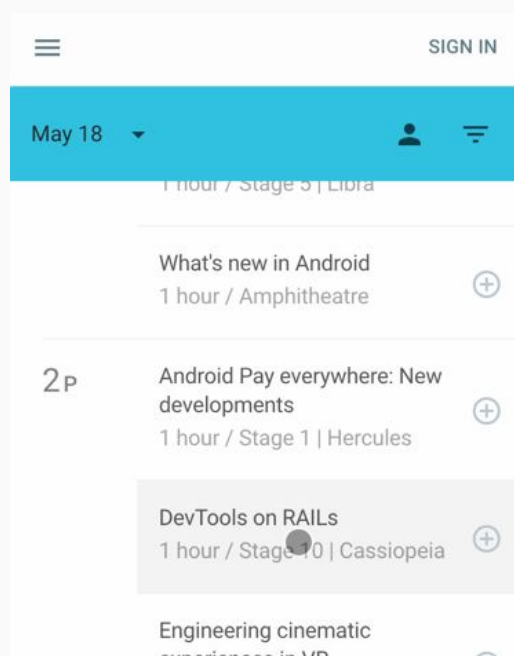
- App-like design
- System fonts
- Non-selectable text in buttons and list items
- Buttons instead of links

2. Fast and fluid transition

- Cache first strategy
- Give a feedback to the user

3. Back arrow should restore the previous scroll position

4. Show taps



App... how much?

Permissions available through the browser



Availability of web permissions



| | | |
|----------------------------|---|---|
| Offline (ServiceWorker.js) | ✓ | ✓ |
| Background sync | ✓ | ✗ |
| Home installation | ✓ | ~ |
| Share (intent) | ✓ | ✗ |
| Foreground detection | ✓ | ✓ |

Availability of web permissions



| | | |
|-----------------------------------------------------|---|---|
| Geolocation | ~ | ~ |
| Accelerometer, Gyroscope, Magnetometer, Orientation | ~ | ~ |
| Bluetooth | ~ | X |
| USB | ✓ | X |
| Payments | ~ | ~ |

Availability of web permissions



| | | |
|------------------------|---|---|
| Browser storage | ✓ | ✓ |
| File access | ✓ | ✓ |
| Network type and speed | ✓ | ✗ |
| Online status | ✓ | ✓ |
| Vibration | ✓ | ✗ |

Availability of web permissions



| | | |
|---------------------------|---|---|
| Battery status | ✓ | ✗ |
| RAM size | ✓ | ✗ |
| Manage clipboard | ~ | ✓ |
| Mouse integration | ✓ | ✓ |
| Presentation (Chromecast) | ✓ | ✗ |

Availability of web permissions



| | | |
|---------------------------|---|---|
| VR | ~ | X |
| Speech recognition | ~ | X |
| Fullscreen | ~ | ~ |
| Screen orientation & lock | ✓ | X |
| Wake lock | ✓ | X |

Coming soon

- NFC
- Contacts
- SMS
- Ambient light
- Task scheduler in SW



What Web Can Do Today

Can I rely on the Web Platform features to build my app?
An overview of the device integration HTML5 APIs

FEATURES

TRAININGS

SERVICES

✓ Feature available in your current browser ✗ Feature not available in your current browser

Camera & Microphone

- 📷 AUDIO & VIDEO CAPTURE ✓
- 📹 ADVANCED CAMERA CONTROLS ✓
- 🎤 RECORDING MEDIA ✓
- 📺 REAL-TIME COMMUNICATION ✓

Native Behaviors

- 📱 LOCAL NOTIFICATIONS ✓
- 📶 PUSH MESSAGES ✓
- 📲 HOME SCREEN INSTALLATION ✓
- 📱 FOREGROUND DETECTION ✓
- 🔒 PERMISSIONS ✓

Seamless Experience

- ⚙️ OFFLINE MODE ✓
- 🔄 BACKGROUND SYNC ✓
- 🔗 INTER-APP COMMUNICATION ✗
- 💳 PAYMENTS ✓
- 🔒 CREDENTIALS ✓

Advertisement

See how your visitors are really using your website

Operating System

Location & Position

How to test

https-localhost + Lighthouse



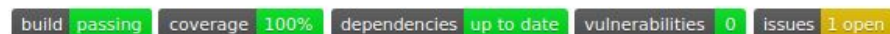
How to test



HTTPS server running on localhost

Run an express server on localhost with HTTP2 and SSL. Serve static files or import as module in your project.

https-localhost is a lightweight tool for serving static content on SSL thanks to locally-trusted development certificates. It works with MacOS, Linux and Windows, on Chrome and Firefox, and requires you no configuration.



Install and use standalone

```
npm i -g https-localhost
```

```
serve ~/myproj
```

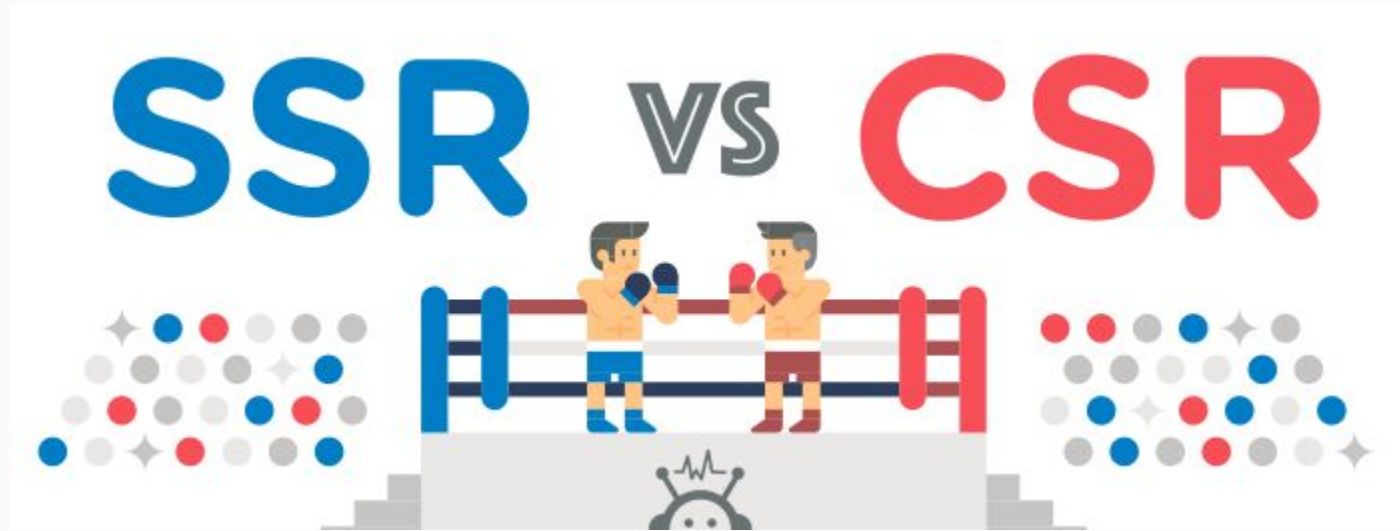
- `sudo` may be necessary.
- If a static path is not provided the current directory content will be served.
- You can change the port setting the PORT environmental variable: `PORT=4433 serve ~/myproj`.

Rendering Strategies

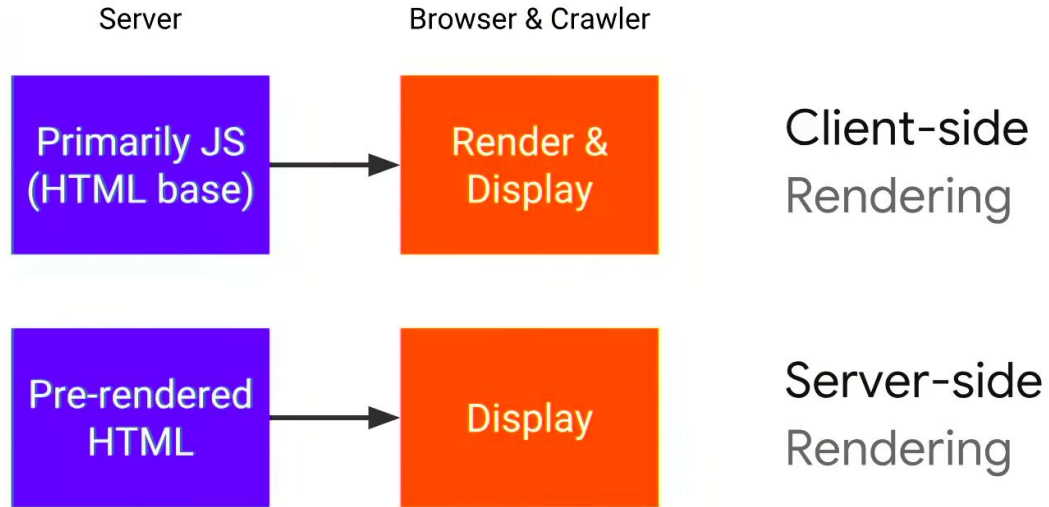
SSR vs CSR vs Hybrid Rendering



Server Side Rendering vs Client Side Rendering

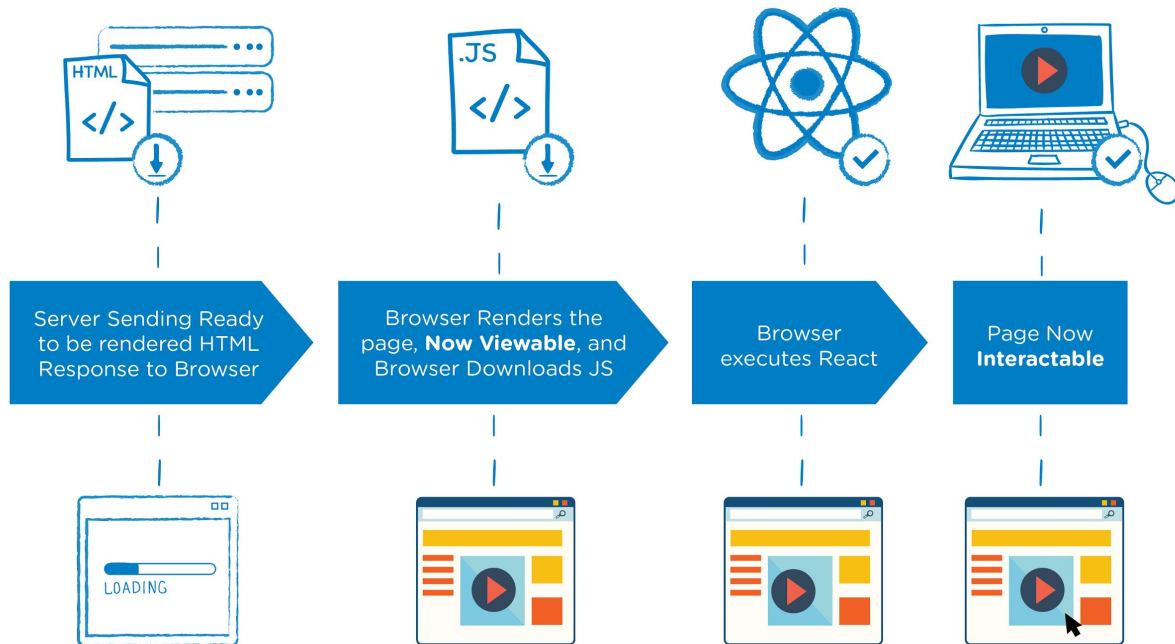


Server Side Rendering vs Client Side Rendering

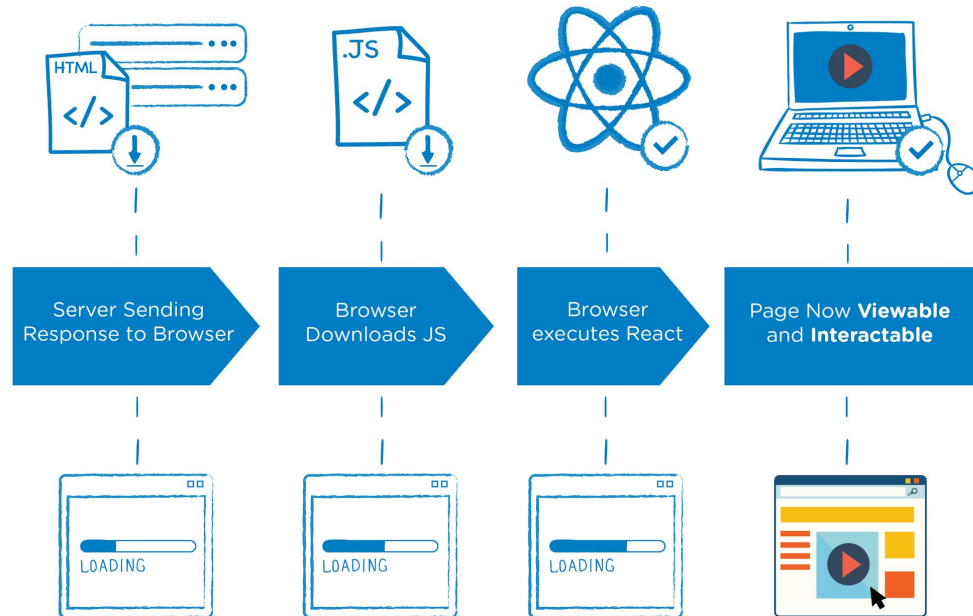


Server Side Rendering

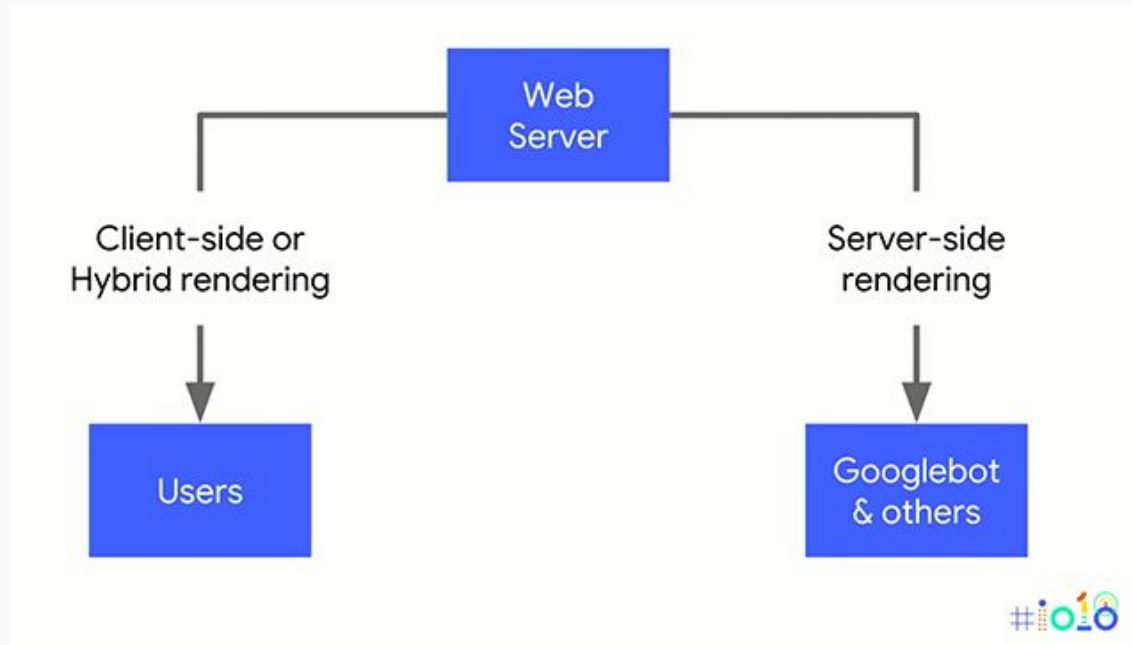
SSR



CSR

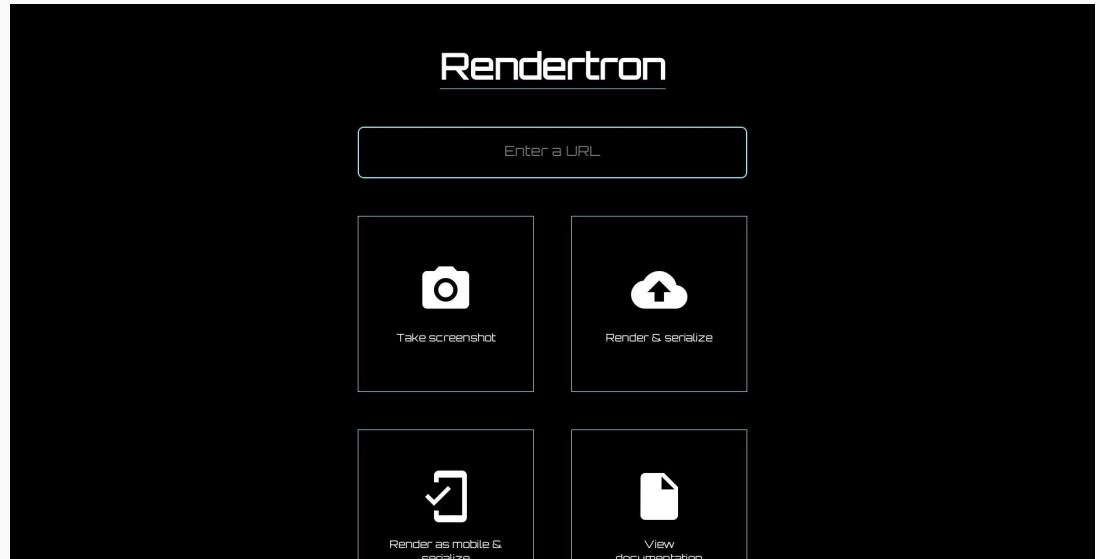
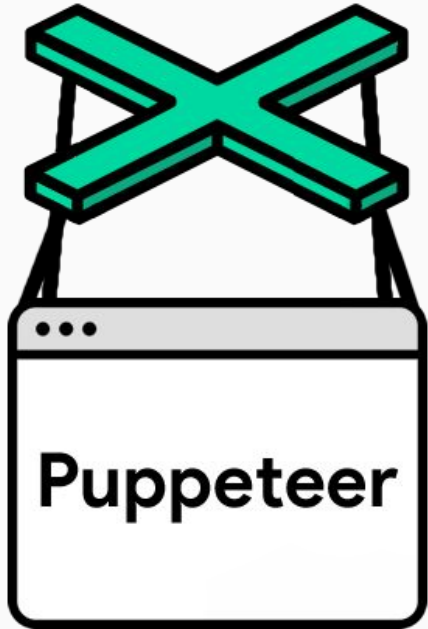


How to crawl a PWA correctly



How to crawl a PWA correctly

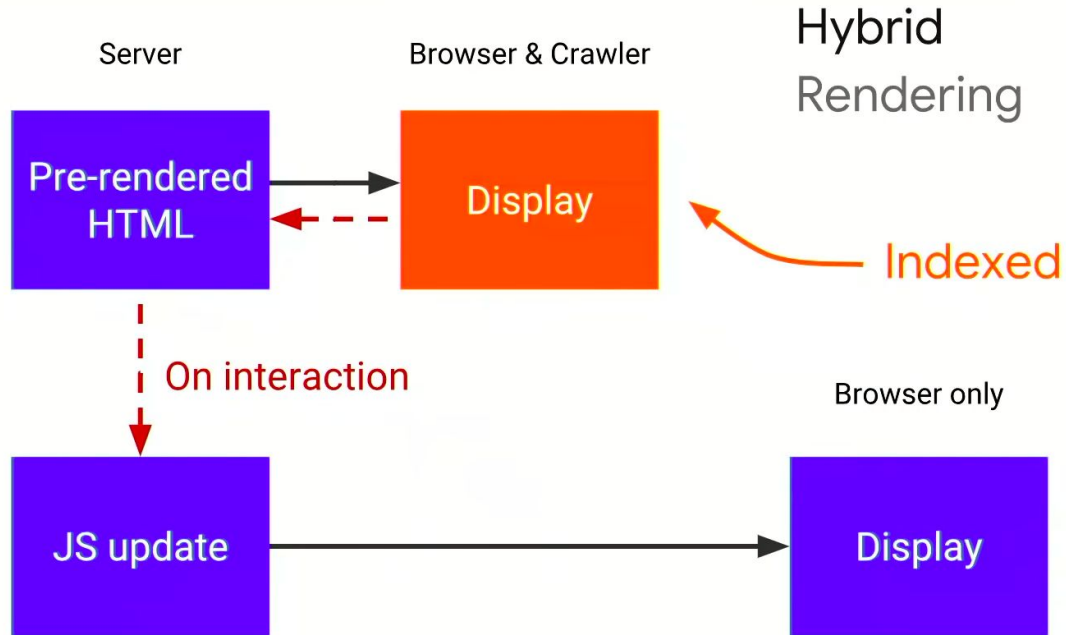
AKA how robots see your website



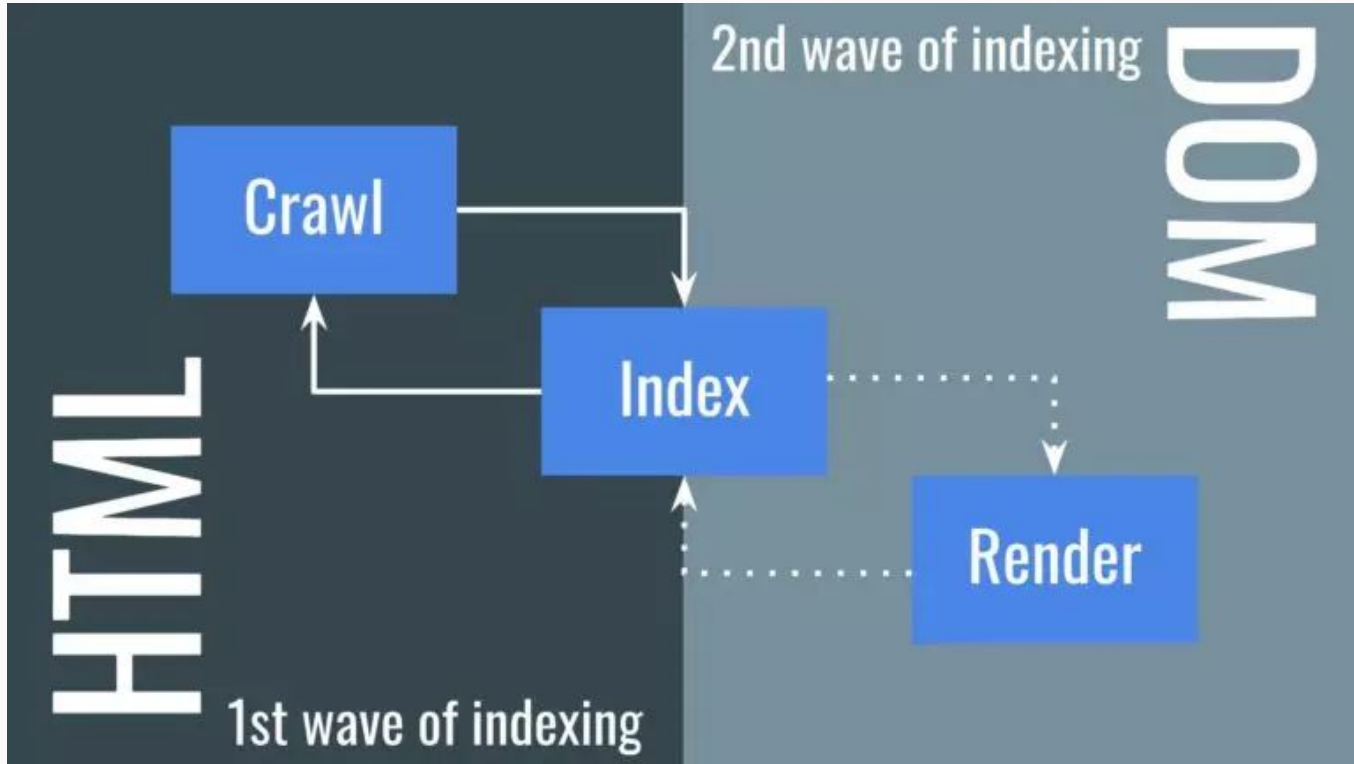
Hybrid rendering

- Server Side Rendering for first page
 - Fast
 - Ready to display
 - Fine for crawlers
- Client Side Rendering for subsequent pages
 - Less data transferred
 - No need for a new app shell
 - If JavaScript is not available use SSR as fallback

Hybrid Rendering



2 waves of indexing

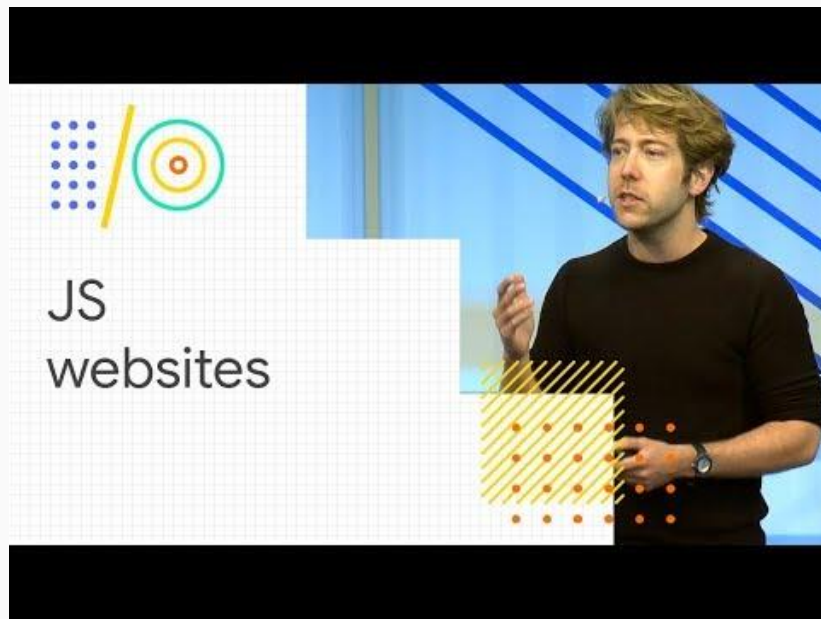


Crawled immediately

Crawled "later" (sometimes)

To deepen the topic

Deliver search-friendly JavaScript-powered websites (Google I/O '18)



Let's code

"Mani in pasta" - cit. grandma'



Let's code

"Mani in pasta" - cit. grandma'

You will need

- An IDE: VSCode or whatever you like
- Node.js
- https-localhost: `https://sserve.dev` or
`npm i -g https-localhost`
- GitHub repository for this workshop: `https://bit.ly/PWArepo` or
`git clone https://github.com/daquinoaldo/pwa-workshop`

Further reading

- [Introduction to Progressive Web App Architectures](#)
Official Google Developers documentation
- [The Offline Cookbook](#)
Google Developers - A curated list of caching strategies examples for the ServiceWorker.js
- [Caching strategies](#)
Vaadin.com - A short and effective overview of caching strategies
- [SEO & Progressive Web Apps: Looking to the Future](#)
Moz blog - SSR, CSR and Hybrid Rendering strategies for a correct crawling
- [SPAs, PWAs and SSR](#)
Marco Otte-Witte, Simplabs - SSR on PWA that are Single Page Applications
- [jsdom](#)
NPM - SSR the ~~wrong~~ fast way
- [Upgrading a create-react-app project to a SSR + code splitting setup](#)
Andrei Duca, Medium - SSR with React, the right way
- [Avoid notches in your PWA with just CSS](#)
Mario Nachbaur, dev.to - The notch problem in PWAs
- [Web Payments Overview](#)
Google Developers - In-PWA payments