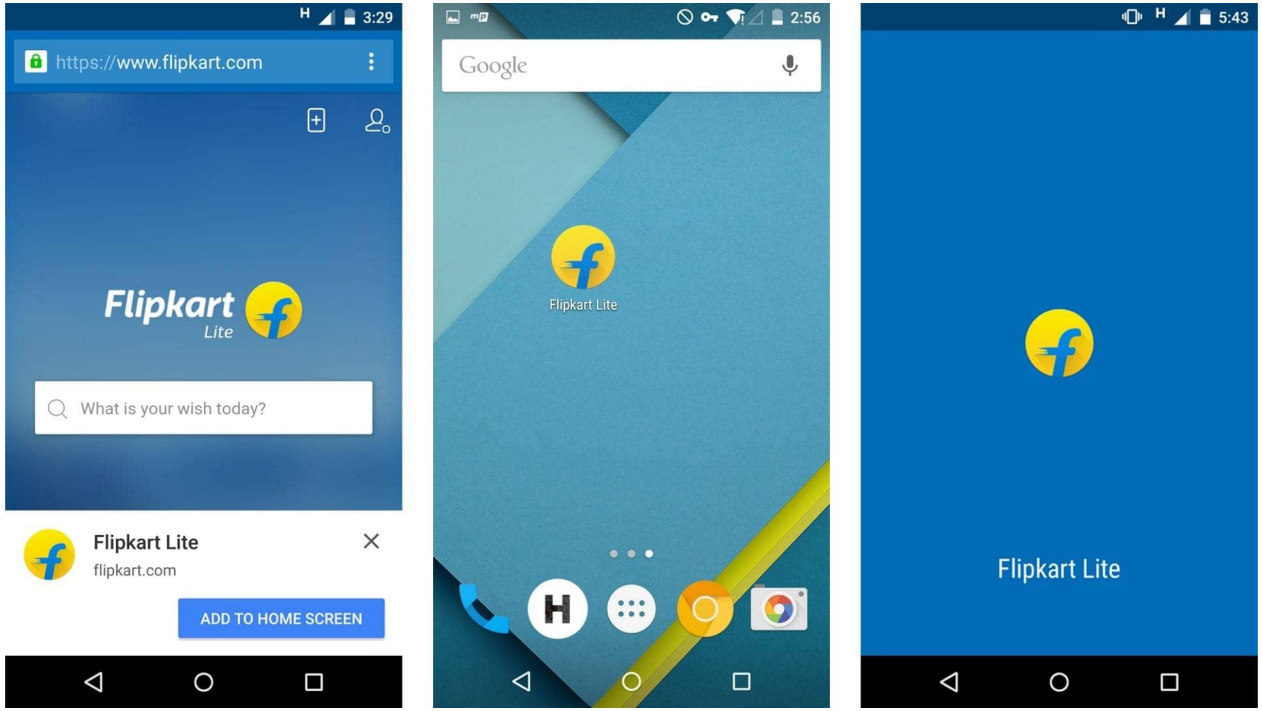
Adaptive
Progressive Web Apps

PWA

> Progressive Web Apps are just great websites that can behave like native apps
>
> Progressive Web Apps are just great apps, powered by Web technologies and delivered with Web infrastructure.

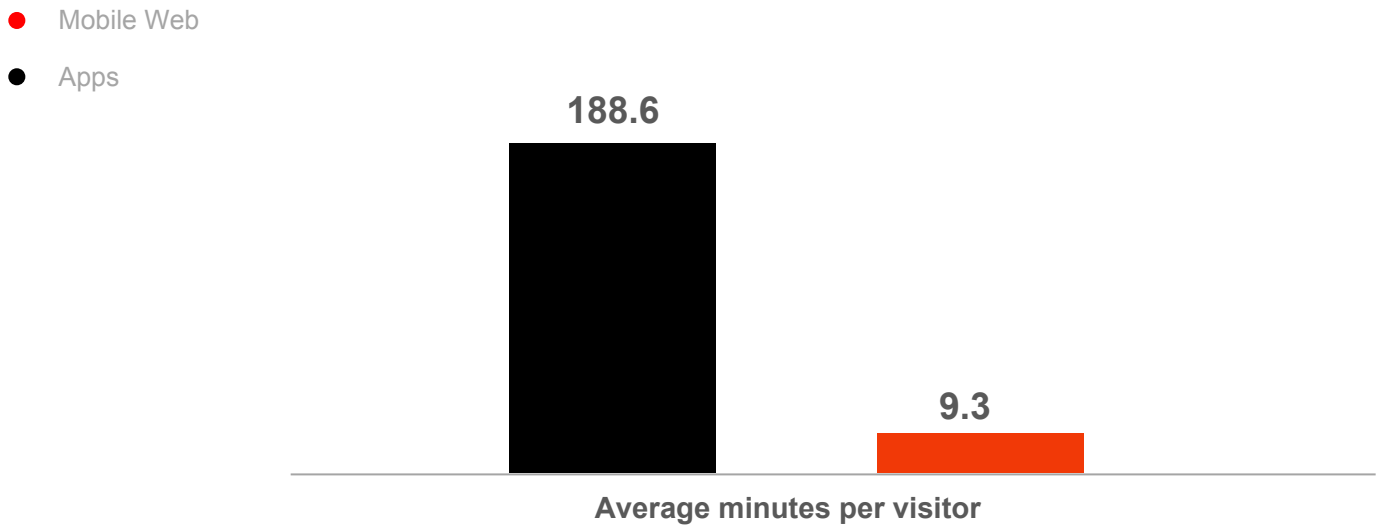
https://bit.ly/2nGa3wH

Mobile PWA’s


3:29
https://www.flipkart.com
Flipkart
Lite
What is your wish today?
Flipkart Lite
flipkart.com
ADD TO HOME SCREEN
2:56
Google
Flipkart Lite
5:43
Flipkart Lite

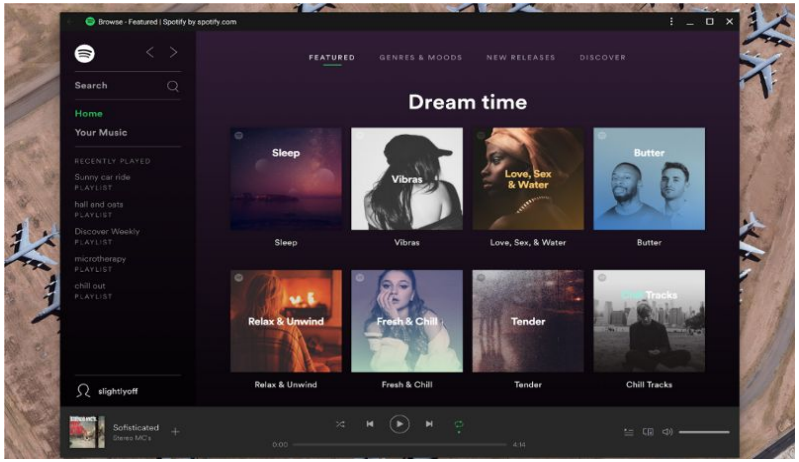
Mobile Web Engagement

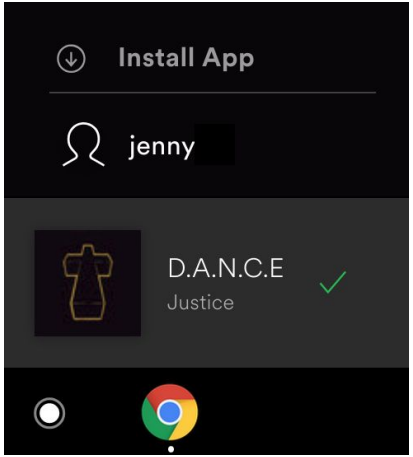
Top 1000 mobile apps vs top 1000 mobile websites

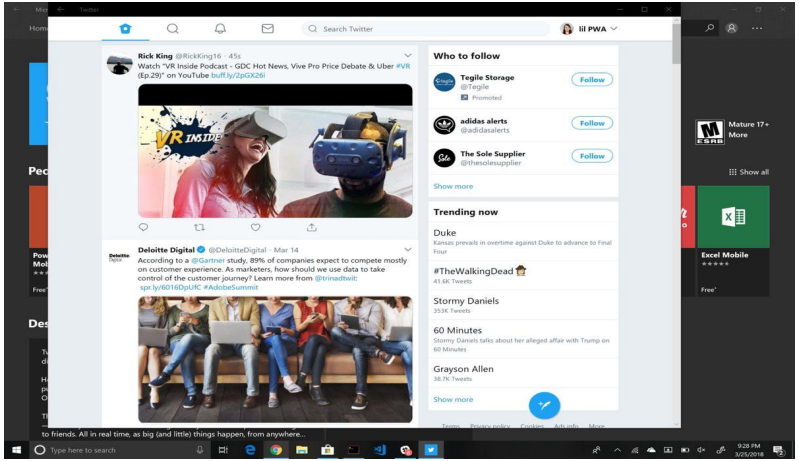


Source: comScore Mobile Metrix, U.S., Age 18+, June 2016

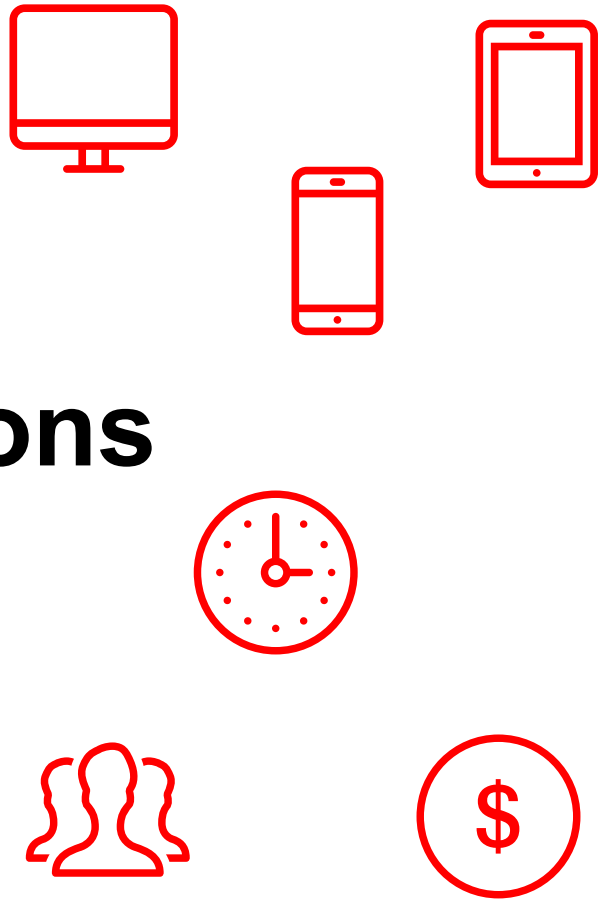
Desktop PWA’s



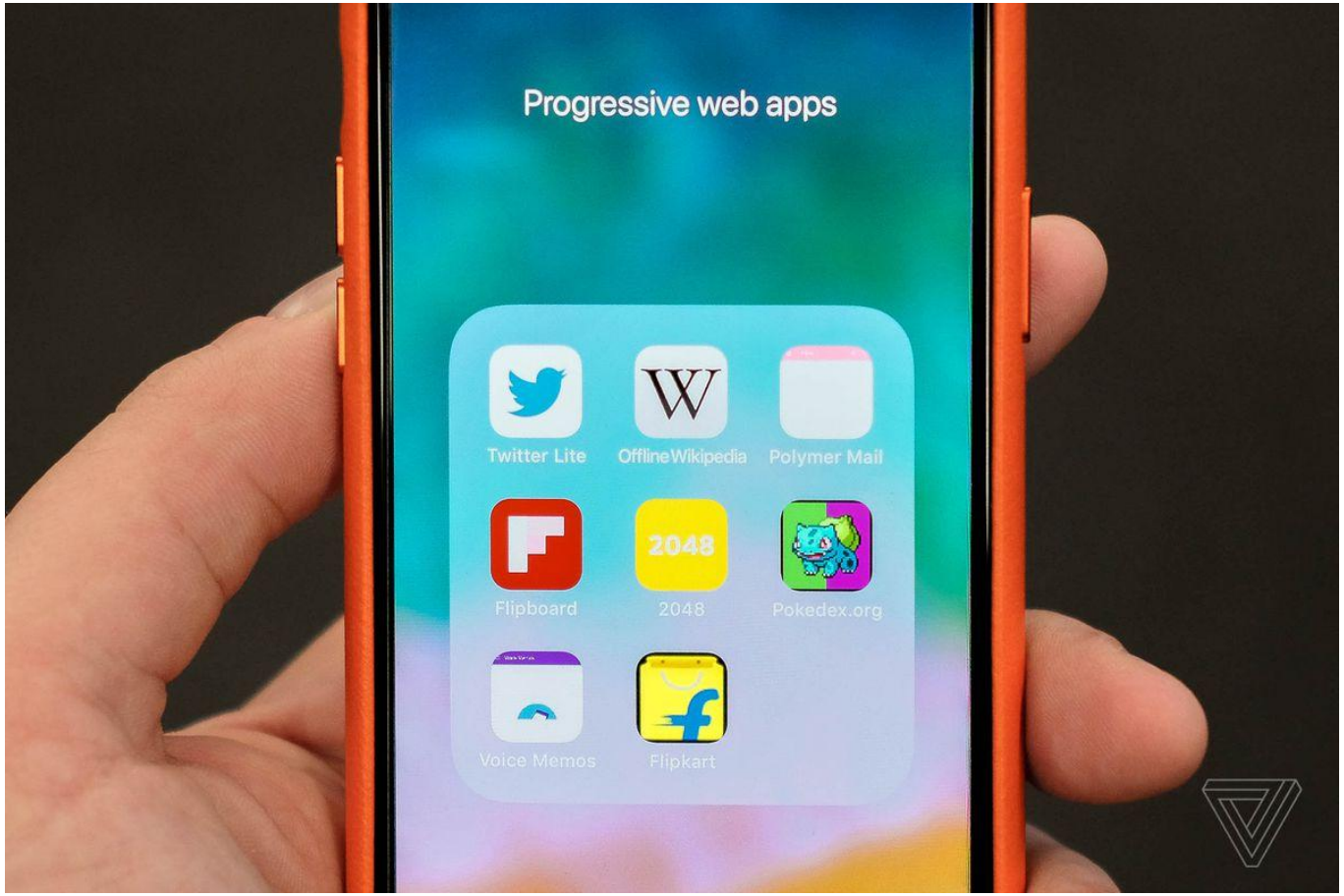




Source: https://developers.google.com/web/updates/2018/05/dpwa



User Expectations


Progressive web apps
Twitter Lite
OfflineWikipedia
Polymer Mail
Flipboard
2048
2048
Pokedex.org
Voice Memos
Flipkart

Design guidelines

Design guidelines for building high quality PWA's

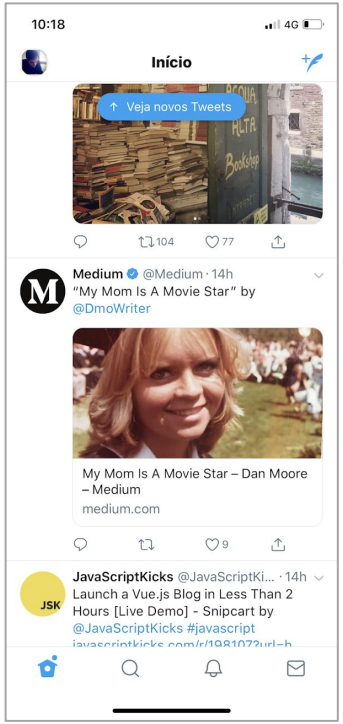
> Start by forgetting everything you know about conventional web design, and instead imagine designing a native app.
>
> Pay attention to detail because native apps have set a precedent for users expectations

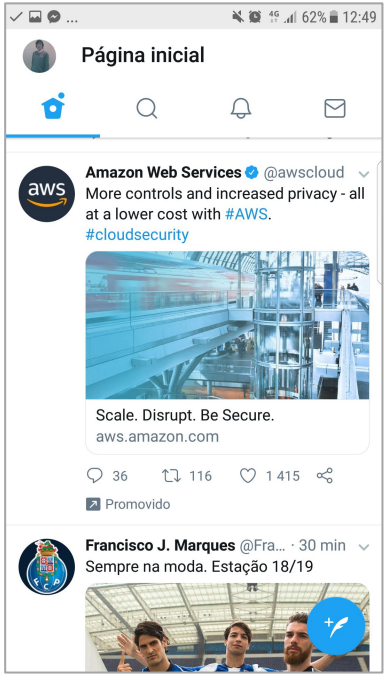
Source: https://developers.google.com/web/ilt/pwa/introduction-to-progressive-web-app-architectures

Navigation Patterns

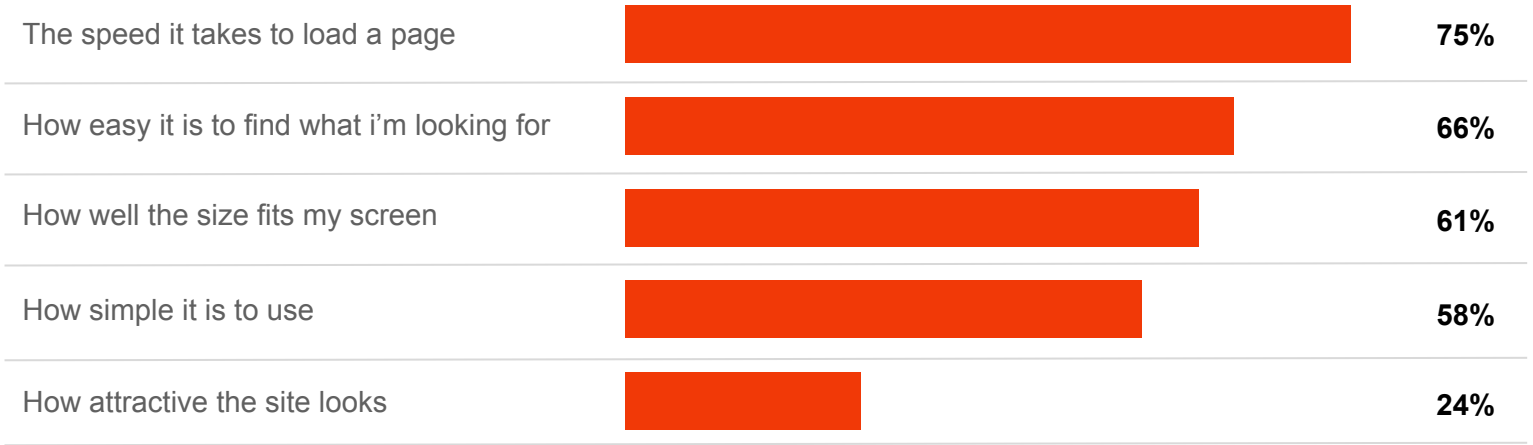
Web

Native





The most important UX feature on mobile is performance



Source: Speed Matters Vol. 3 - https://bit.ly/2Ldo8fs

Device & Context

On the move things feel slower

% of user that perceived a website to have loaded relatively fast

75%

Sat Down

52%

On the move

When you need the information
ASAP things feel slower

Source: Speed Matters Vol. 3 - https://bit.ly/2Ldo8fs

Device & Context

Users expectations differ depending on their current context or device

Mobile

Lightweight

Performant

Fast

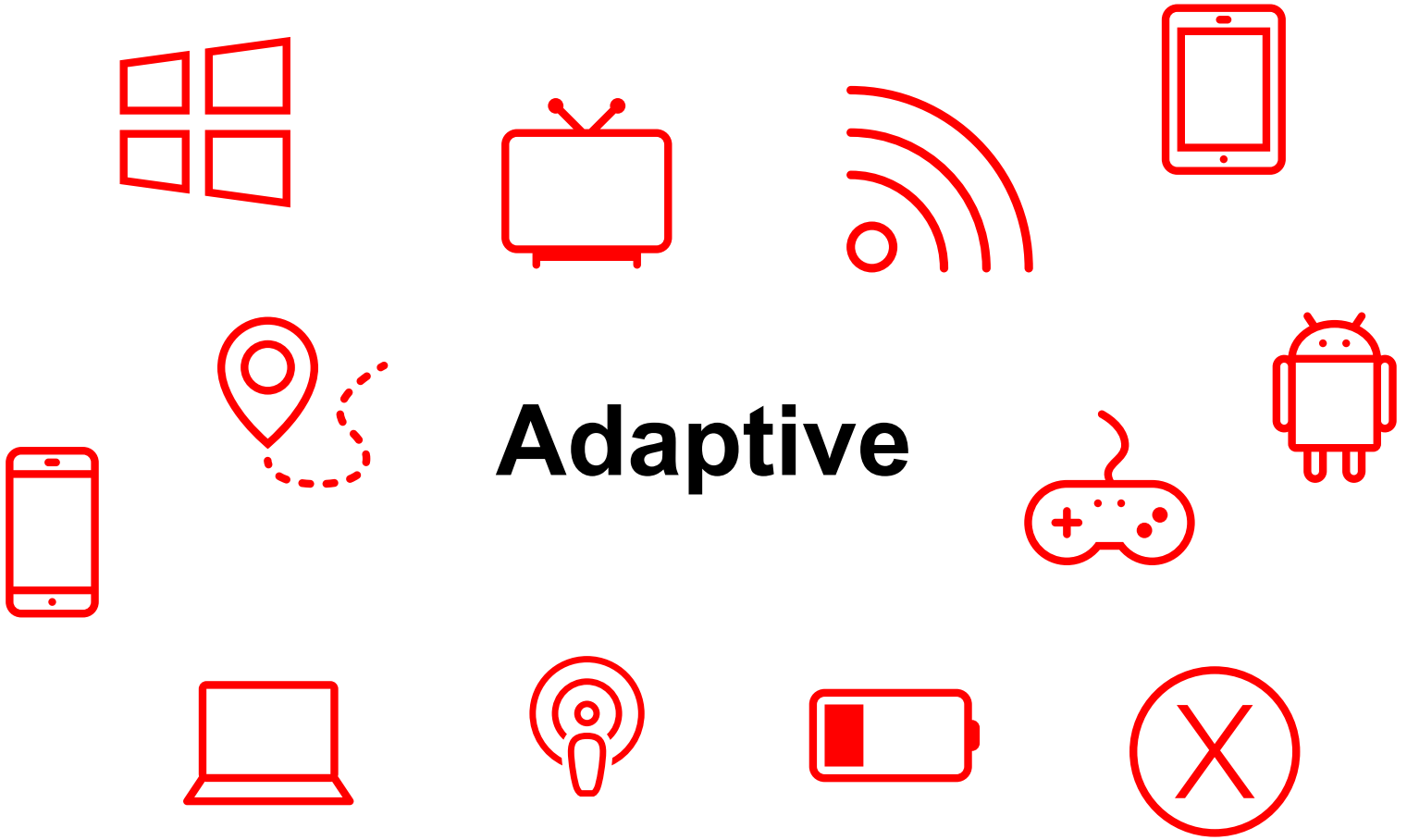
Desktop

Immersive

Complementary

Convenient

Source: https://developer.microsoft.com/en-us/events/build/content/modernizing-twitter-for-windows-as-a-pwa


Adaptive

Adaptive

Adapting to user device and context

One codebase, One URL

Adaptive & Responsive

Differentiated & optimal experience based on available information

Adaptation Properties

Static

- Operative System
- Device Memory
- Feature Support

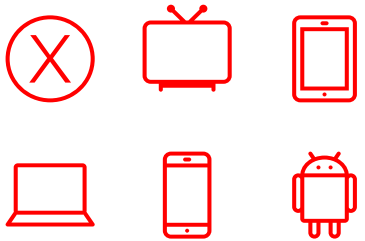
Dynamic

- Quality of connection
- Location
- Available storage
- User preferences

We’ll build an application on top of three concepts:

1. **Targeted builds**
2. **Smart-start**
3. **Capability reporting**

Targeted Builds



- android.js
- ios.js
- desktop.js
- es6.js
- lite.js

Targeted Builds

```
const createVariants = require('parallel-webpack').createVariants;
const webpackConfig = require('./webpack.client.config');

const buildVariants = {
    capability: ['modern', 'legacy'],
    platform: ['default', 'ios', 'android'],
};

const createConfig = options => {
    return webpackConfig(options);
};

module.exports = createVariants({}, buildVariants, createConfig);
```

Targeted Builds

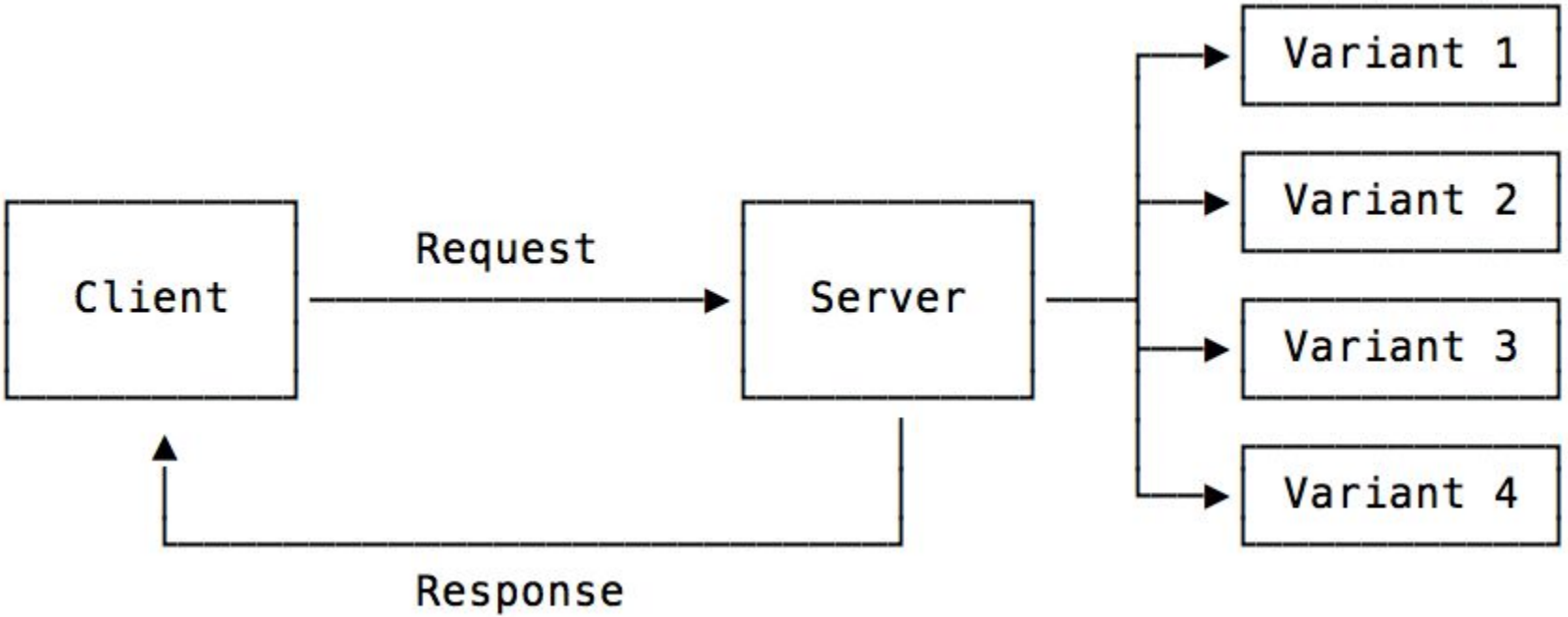
Use cases

Bundle only necessary polyfills based on client browser support

Adjust javascript transpilation

Bundle different components based on build parameters

Smart-Start


Client
Request
Server
Variant 1
Variant 2
Variant 3
Variant 4
Response

Smart-start

```
app.get('/', async (req, res) => {
    const agent =
useragent.parse(req.headers['user-agent']);
    const capabilities = {
        'browser': agent.family,
        'browserVersion': agent.major,
        'OS': agent.os.family,
        'OSVersion': agent.os.major,
    };
    const capability = capabilityFilter(capabilities);
    const platform = platformFilter(capabilities);

    // Create file path based on negotiation results
    const variant = `${platform}.${capability}.js`;

    res.send(variant);
});
```

Smart-start

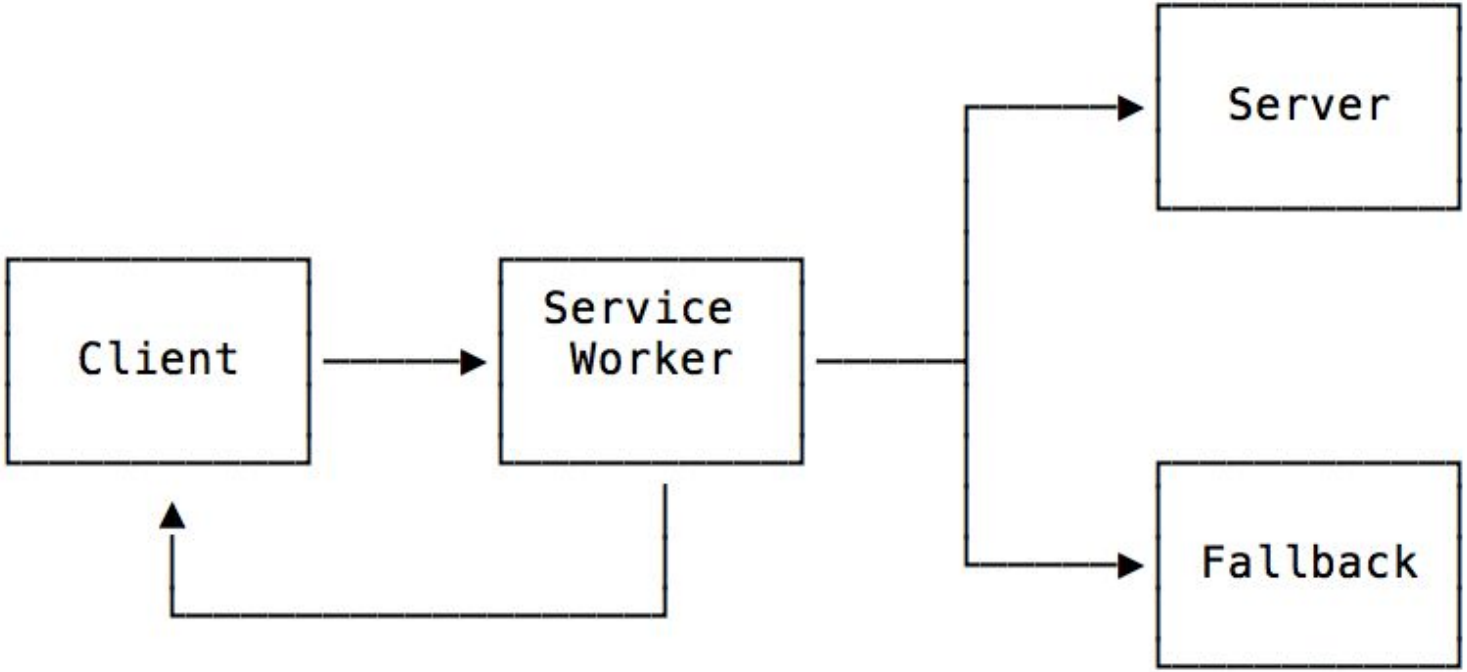
Use cases

Serve different targeted builds

Adjust API responses

Capability Reporting

Service-worker



Capability Reporting

```
"use strict";
// Listen to fetch events
self.addEventListener('fetch', function(event) {
    //Check for network connection speed
    const networkSpeed = navigator.connection.downlinkMax
    // Add extra header to request
    event.respondWith(
        fetch(`${req.url}`,{
            headers: {
                'Content-Type': 'image/svg+xml',
                'downlinkMax': networkSpeed,
            },
        })
    )
});
```

Capability Reporting

Use cases

Enhance requests with information available on the client

Selectively load images

Provide request fallbacks

Define Request SLA´s

Demo

Adaptive

Further Use cases

Cache different sets of assets depending on the user device storage capacity

Reduce or completely halt 3rd party loading

Disable high dpi image loading if the connection is slow let the user selectively choose which images he wants to see in high resolution

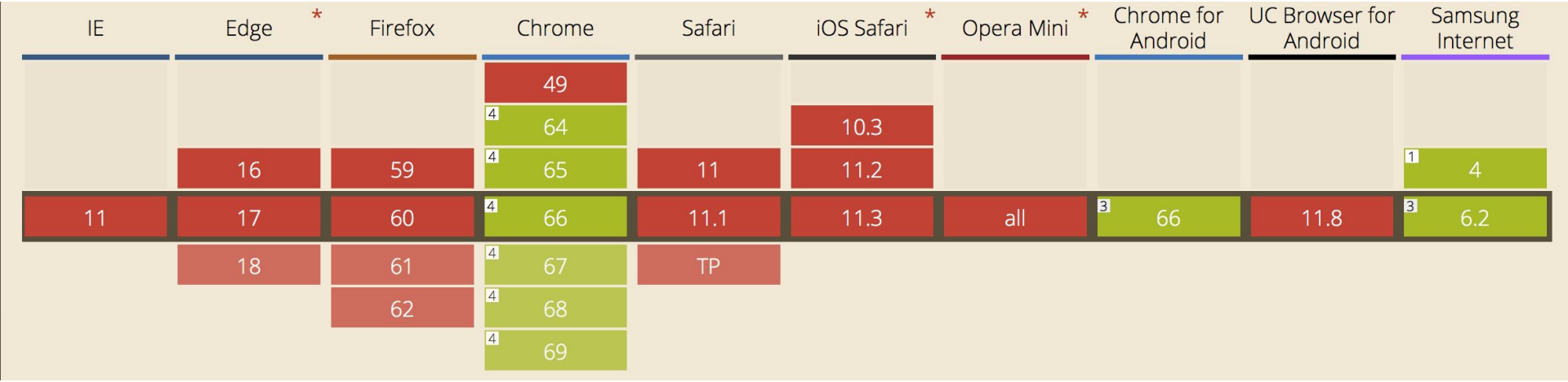
Browser support

Service Worker

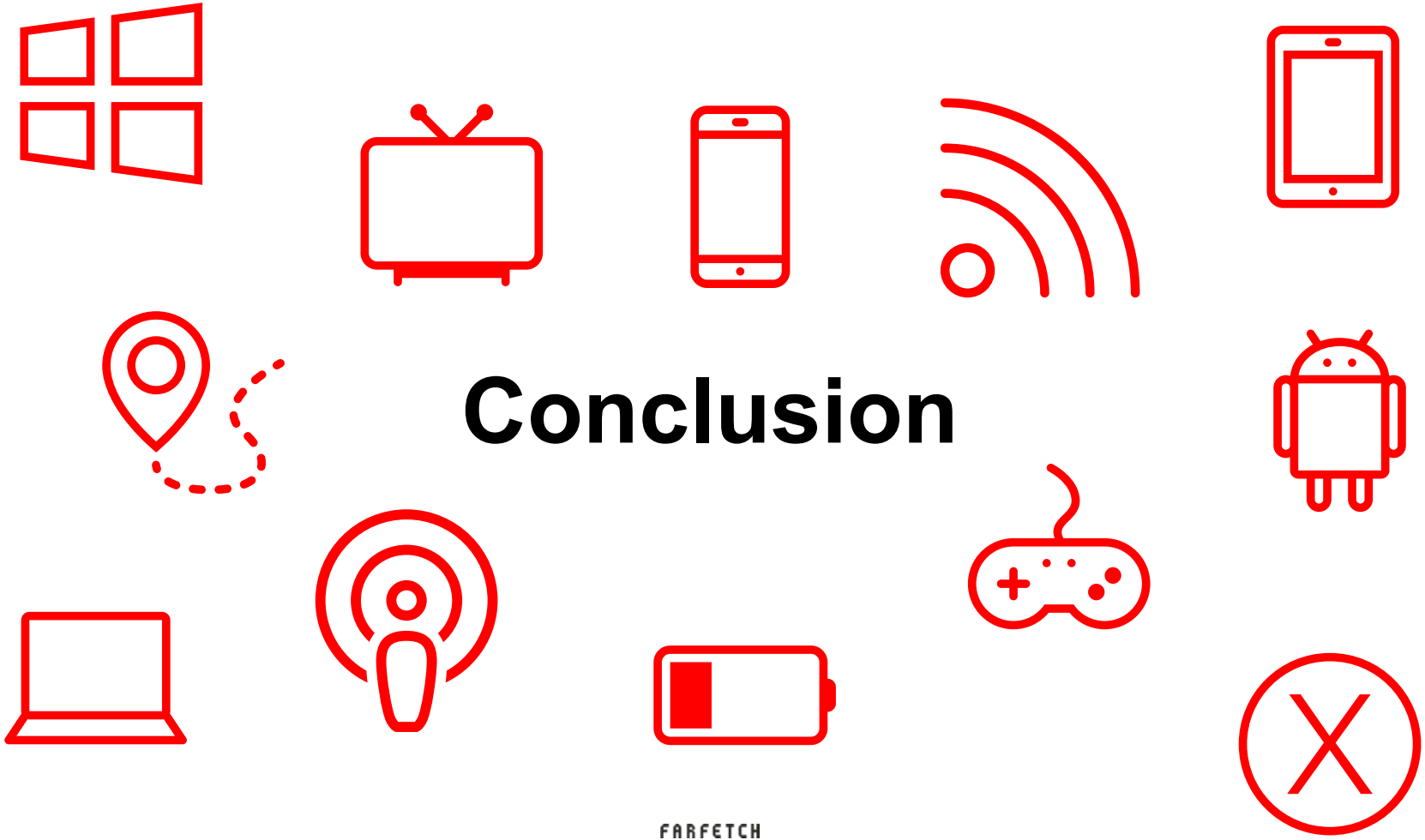
IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			49						
			64		10.3				
	[2] 16	59	65	11	11.2				4
11	17	60	66	11.1	11.3	all	66	11.8	6.2
	18	61	67	TP					
		62	68						
			69						

Browser support

Network Information



Source: https://caniuse.com/#feat=netinfo



Conclusion

Conclusion

Adaptive PWA’s

Start with a great base PWA and then account for adaptation parameters

Think about multiple devices, platforms, contexts and adaptation strategies

Ultimately it’s all about providing a great user experience.

Thank you

@luisvieira_gmr

luis.vieira@farfetch.com