PWA-开发如App般体验的Web应用

前端开发部-研发一组 冯伟平

2017.7

JD.京东.COM

为什么？

JD.COM 京东

Why?

为什么?

JD.COM 京东

App是什么样的?

1. 清一色的摆在主屏幕上
2. 没有“白屏”现象
3. 不依赖浏览器
4. 不需要记住网址

有没有一种办法让web应用和app一样呢?

为什么？

JD.COM 京东

Web应用没有优势吗？！

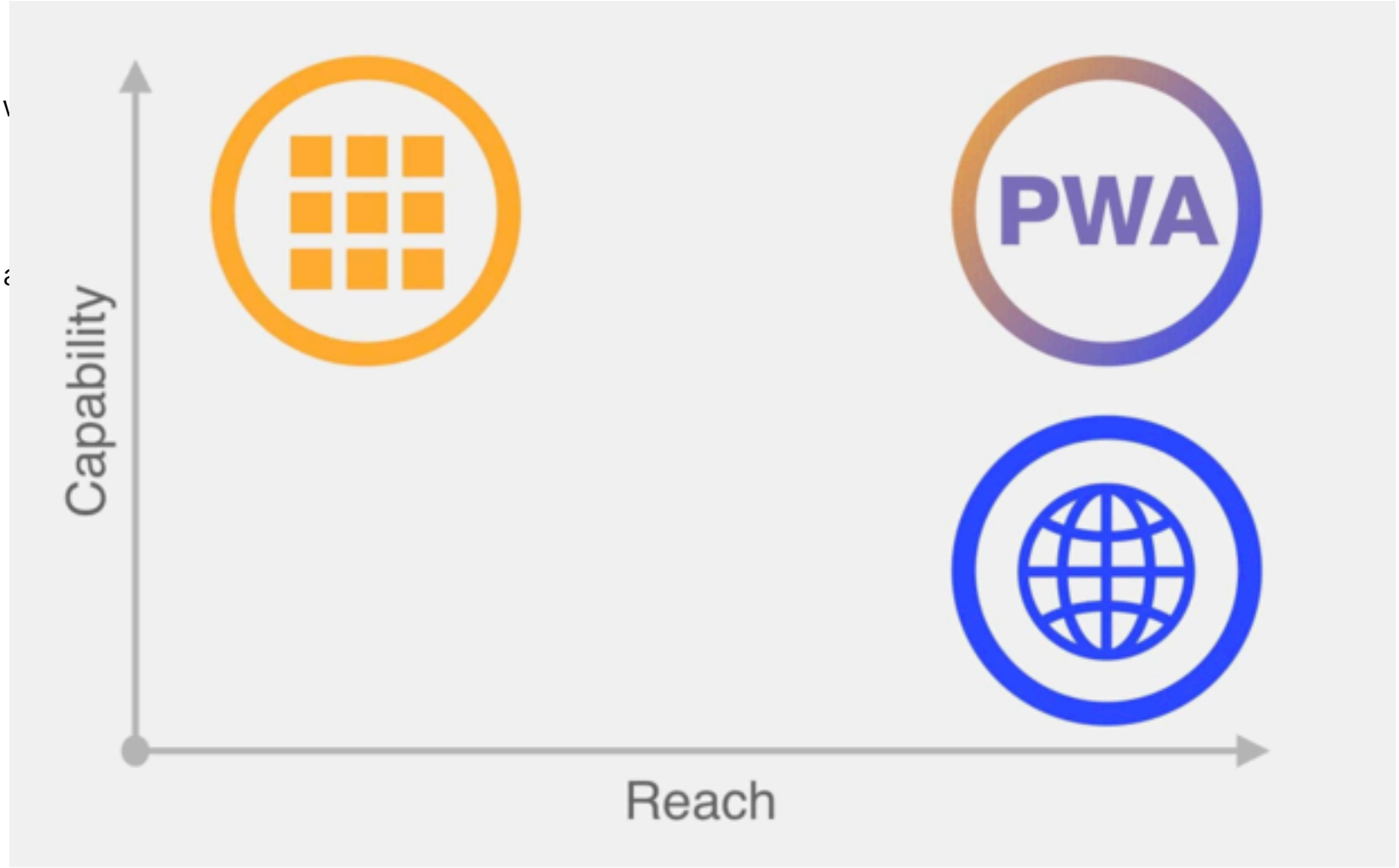
1. 免于安装
2. 随叫随到，用完即走
3. 无需更新
4. 易于传播分享
5. 发布简单

有没有类似app的应用，可以保留这些优势？(不是小程序)

为什么?

JD.COM 京东

鱼与熊掌不可兼得吗?



PWA是什么?

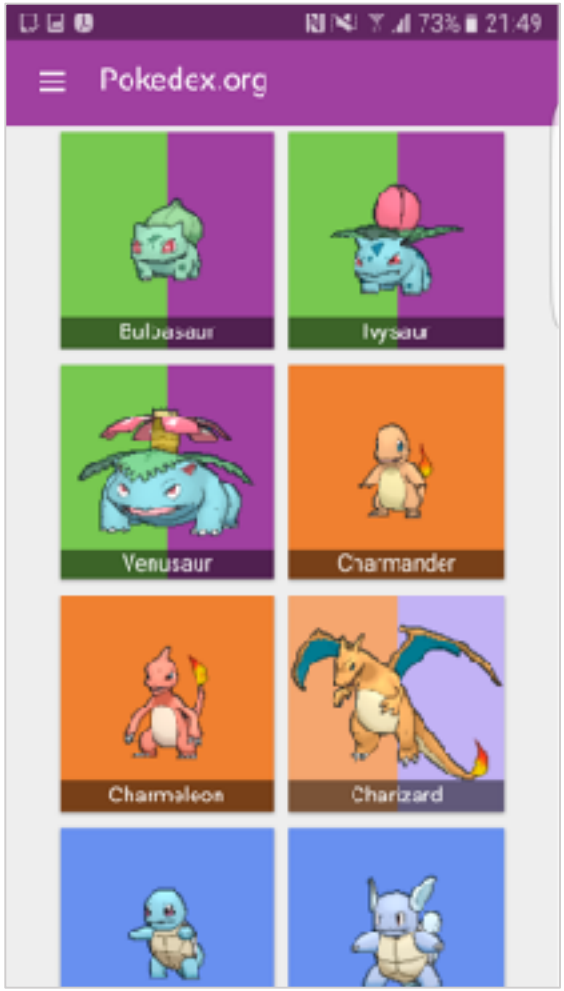
JD.COM 京东

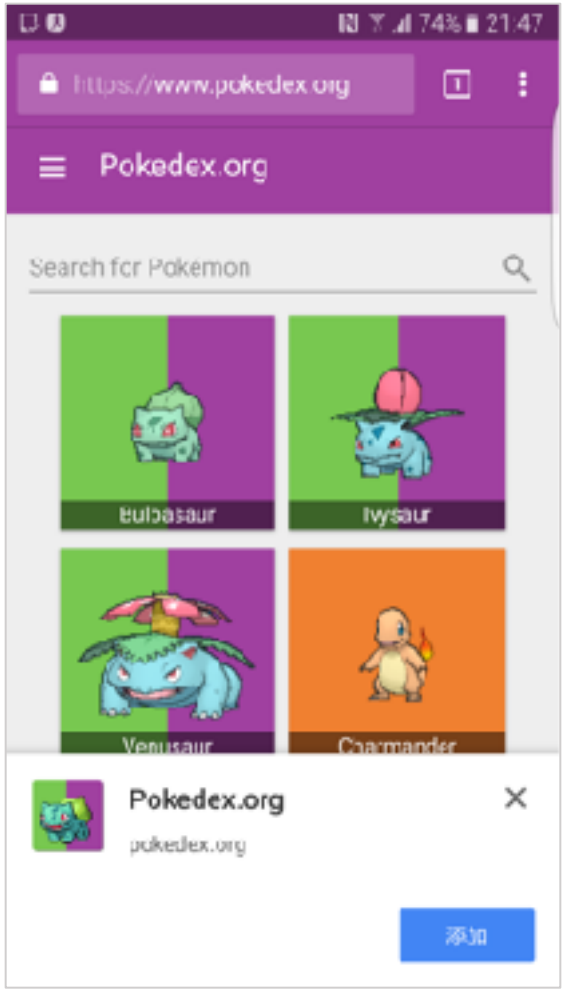
What?

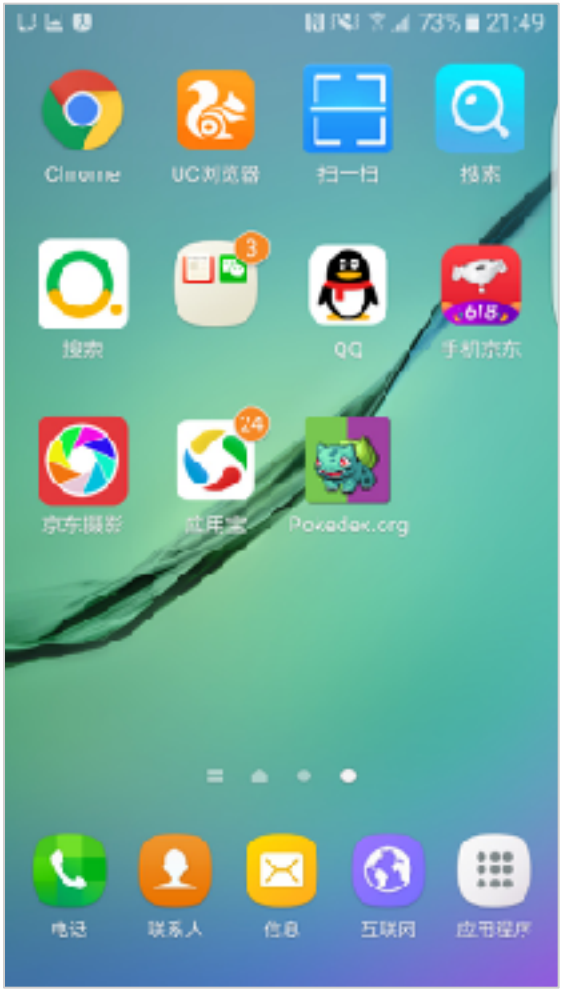
认识PWA

JD.COM 京东

https://pwa.rocks 有诸多PWA的实践应用



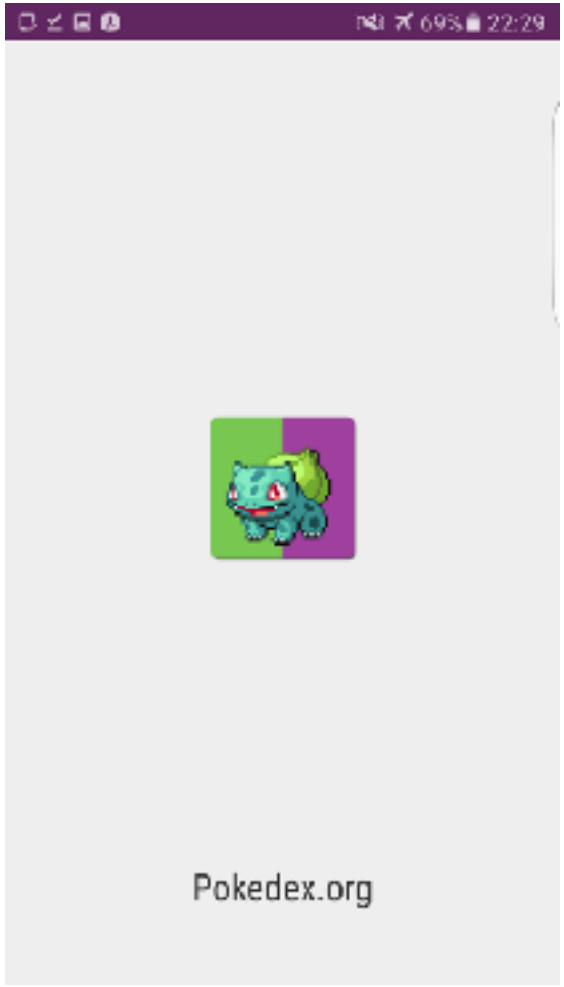


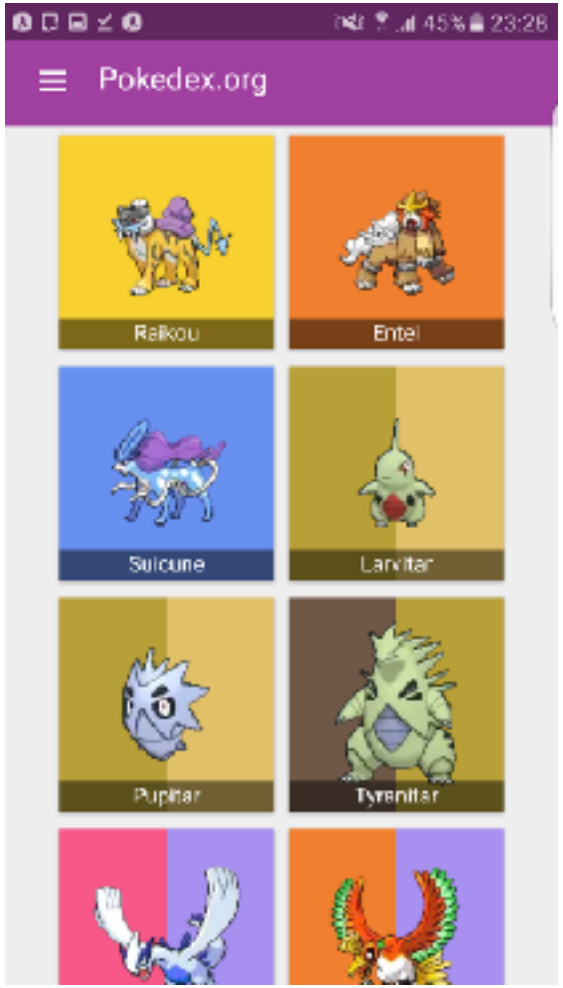


认识PWA

JD.COM 京东

离线状态下





JD.COM 京东

什么是PWA?

1. PWA （ Progressive Web Apps，渐进式网页应用）是由谷歌提出的新一代 Web 应用概念，旨在提供可靠、快速、类似 Native 应用的服务方案。

2. 在安全、性能和体验三个方面都有很大提升；

3. PWA 本质上是 Web App，借助一些新技术也具备了 Native App 的一些特性，兼具 Web App 和 Native App 的优点；

Progressive Web App Checklist

认识PWA

JD.COM 京东

特性:

1. **渐进式 -** 适用于所有浏览器，因为它是以渐进式增强作为宗旨开发的
2. **连接无关性 -** 借助Service Worker在离线或者网络较差的情况下正常访问
3. **类似应用 -** 由于是在 App Shell 模型基础上开发，因为应具有 Native App 的交互和导航，给用户 Native App 的体验
4. **持续更新 -** 始终是最新的，无版本和更新问题
5. **安全 -** 通过 HTTPS 协议提供服务，防止窥探和确保内容不被篡改
6. **可索引 -** 应用清单文件和 Service Worker 可以让搜索引擎索引到，从而将其识别为『应用』
7. **粘性 -** 通过推动离线通知等，可以让用户回流
8. **可安装 -** 用户可以添加常用的 webapp 到桌面，免去去应用商店下载的麻烦
9. **可链接 -** 通过链接即可分享内容，无需下载安装

关键技术：

1. Service Worker & Cache Storage： 能够显著提高应用加载速度、甚至让 web 应用可以在离线环境使用；
2. App Manifest： 描述 web 应用元数据（metadata）、让 web 应用能够像原生应用一样被添加到主屏、全屏执行；
3. Notifications &Push API： 提高 web 应用与操作系统集成能力，让 web 应用能在未被激活时发起推送通知；

Service Worker

JD.COM 京东

Service Worker 是PWA的核心技术

W3C 组织早在 2014 年 5 月就提出过 Service Worker 这样的一个 HTML5 API

https://developers.google.cn/web/fundamentals/getting-started/primers/service-workers

Service Worker

JD.COM 京东

Service Worker 有以下功能和特性：

1. 一个独立的 worker 线程，独立于当前网页进程，有自己独立的 worker context。

2. 一旦被 install，就永远存在，除非被 uninstall

3. 需要的时候可以直接唤醒，不需要的时候自动睡眠（有效利用资源，此处有坑）

4. 可编程拦截代理请求和返回，缓存文件，缓存的文件可以被网页进程取到（包括网络离线状态）

5. 离线内容开发者可控

6. 能向客户端推送消息

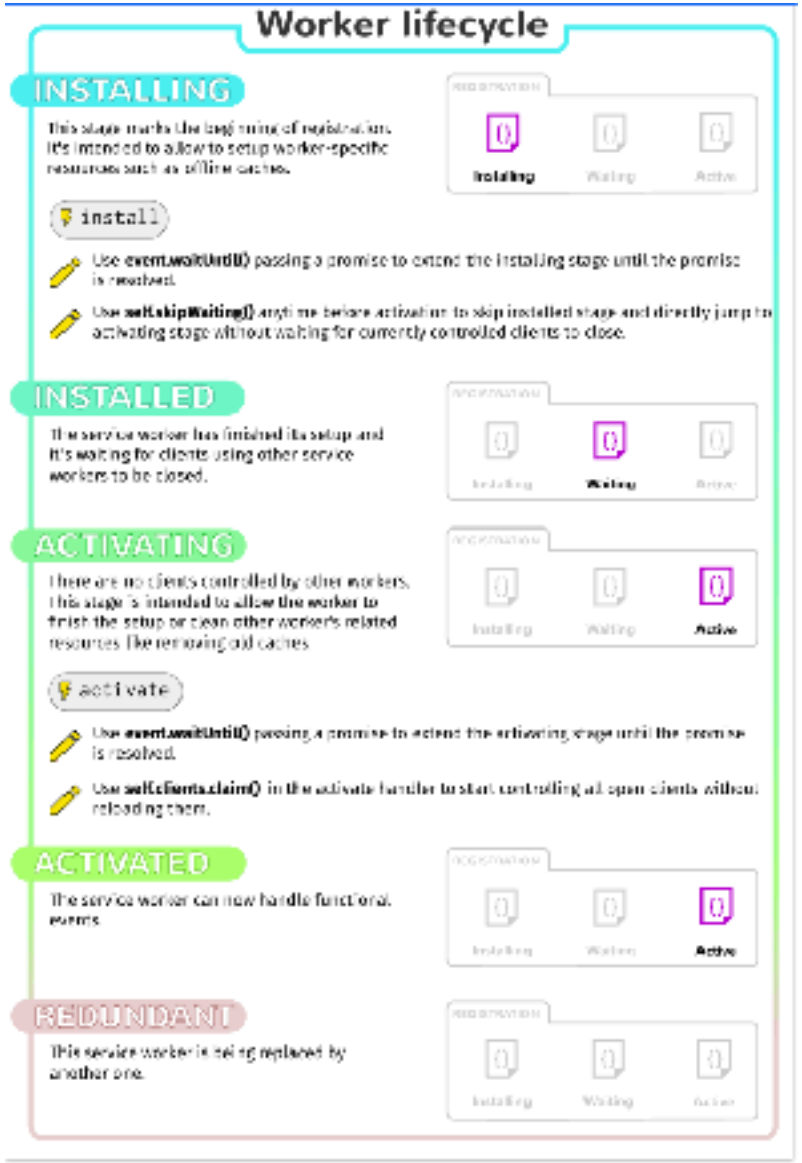
7. 不能直接操作 DOM

8. 出于安全的考虑，必须在 HTTPS 环境下才能工作

9. 异步实现，内部大都是通过 Promise 实现

Service Worker

JD.COM 京东



- **安装中:**触发 install 事件回调指定一些静态资源进行离线缓存
- **安装后:**Service Worker 已经完成了安装，并且等待其他的 Service Worker 线程被关闭
- **激活中:**触发activate 回调、event.waitUntil()、self.clients.claim()
- **激活后:**处理功能性的事件 fetch (请求)、sync (后台同步)、push (推送)
- **废弃:**一个 Service Worker 的生命周期结束

Service Worker

JD.COM 京东

注册及注销

```
3  //ServiceWorkerService.js
4  const SERVICE_WORKER_FILE_PATH = 'sw.js';
5  const SW = navigator.serviceWorker;
6  const isSupportServiceWorker = () => SERVICE_WORKER_API in navigator;
7  const sendMessageToSW = msg => new Promise((resolve, reject) => {...
18 });
19 if (isSupportServiceWorker()) {
20   if(SW_SWITCH){
21     window.addEventListener('load', function () {
22       var _sw = SW.register(SERVICE_WORKER_FILE_PATH)
23         .then((registration) => console.log('ServiceWorker registration successful'))
24         .catch((err) => console.error('ServiceWorker registration failed: ', err))
25         .then(() => sendMessageToSW('Hello!, service worker.'))
26         .catch(() => console.error('Send message error.'));
27     })
28   }else{
29     SW.ready.then(reg => {
30       reg.unregister();
31     })
32   }
33 } else {
34   console.info('Browser not support Service Worker.');
35 }
```

降级开关SW_SWITCH 可在头尾系统及CMS系统配置

Service Worker

JD.COM 京东

静态缓存设置

```
1  //sw.js
2  const HOST_NAME = location.host;
3  const VERSION_NAME = 'CACHE-006';
4  const CACHE_NAME = HOST_NAME + '-' + VERSION_NAME;
```

```
77  //sw.js
78  const cacheList = [
79    "index.html",
80    "/css/whatever-v3.css",
81    "/js/all-min-v4.js"
82  ]
83  const onInstall = function(e){
84      e.waitUntil(
85          caches.open(CACHE_NAME).then(function(cache) {
86              console.log('Adding to Cache:', cacheList)
87              return cache.addAll(cacheList)
88          }).then(function() {
89              console.log('Skip waiting!')
90              return self.skipWaiting()
91          })
92      )
93  }
94  self.addEventListener('install', onInstall)
```

大文件不要在此处设置缓存，影响sw注册

Service Worker

JD.COM 京东

Service Worker更新

```
1  //sw.js
2  const HOST_NAME = location.host;
3  const VERSION_NAME = 'CACHE-006';
4  const CACHE_NAME = HOST_NAME + '-' + VERSION_NAME;
```

```
44 const onActivate = function(event) {
45   event.waitUntil(
46     Promise.all([
47       caches.keys().then(cacheNames => {
48         return cacheNames.map(name => {
49           if (name !== CACHE_NAME) {
50             sentMessage('Update cache ');
51             return caches.delete(name)
52           }
53         })
54       })
55     ])
56     .then(() => {
57       console.log('Clients claims.')
58       return self.clients.claim()
59     })
60   )
61 }
62 self.addEventListener('activate', onActivate)
```

Service Worker

JD.COM 京东

动态缓存设置

```
117  const handleFetchRequest = function(req) {
118      if (isNeedCache(req.url)) {
119          const request = isCORSRequest(req.url, HOST_NAME) ? new Request(req.url, {mode: 'cors'}) : req;
120          return caches.match(request)
121              .then(function(response) {
122                  let fetchRequest = request.clone();
123                  if (response) {
124                    fetch(fetchRequest)
133                    return response;
134                  }
135                  return fetch(fetchRequest)
136                      .then(function(response) {
137                        if(!isValidResponse(response)) return response;
138                        const clonedResponse = response.clone();
139                        caches.open(CACHE_NAME)
140                          .then(function(cache) {
141                              cache.put(request, clonedResponse);
142                          });
143                        return response;
144                      });
145              });
146      } else {
147          return fetch(req);
148      }
149  };
150  const onFetch = function(event) {
151      event.respondWith(handleFetchRequest(event.request));
152  };
153  self.addEventListener('fetch', onFetch)
```

跨域请求需要服务端支持cors"跨域资源共享"

Service Worker

JD.COM 京东

Service Worker 错误监控

```
124  self.addEventListener('error', event => {
125    // 上报错误信息
126    // 常用的属性:
127    // event.message
128    // event.filename
129    // event.lineno
130    // event.colno
131    // event.error.stack
132  })
133  self.addEventListener('unhandledrejection', event => {
134    // 上报错误信息
135    // 常用的属性:
136    // event.reason
137  })
```

1. SW 大部分 API 都是 promise-based 的，promise 里未处理的错误触发的不是 error事件，而是 unhandledrejection 事件
2. 这两个事件都只能在 worker 线程的 initial 生命周期里注册

App Manifest

JD.COM 京东

App Manifest: 用于描述 web 应用元数据（metadata）、让 web 应用能够像原生应用一样被添加到主屏、全屏执行；

1. 不优雅
2. 重复定义
3. 可读性差
4. 难维护
5. 与 html 耦合

<meta>/<link> 标签来为 web 应用添加

```
<!-- Add to homescreen for Safari on iOS -->
<meta name="apple-mobile-web-app-capable" content="yes">
<meta name="apple-mobile-web-app-status-bar-style" content="black">
<meta name="apple-mobile-web-app-title" content="Lighten">

<!-- Add to homescreen for Chrome on Android -->
<meta name="mobile-web-app-capable" content="yes">
<mate name="theme-color" content="#000000">

<!-- Icons for iOS and Android Chrome M31-M38 -->
<link rel="apple-touch-icon-precomposed" sizes="144x144" href="images/touch/apple-touch-i
<link rel="apple-touch-icon-precomposed" sizes="114x114" href="images/touch/apple-touch-i
<link rel="apple-touch-icon-precomposed" sizes="72x72" href="images/touch/apple-touch-ico
<link rel="apple-touch-icon-precomposed" href="images/touch/apple-touch-icon-57x57-precon

<!-- Icon for Android Chrome, recommended -->
<link rel="shortcut icon" sizes="196x196" href="images/touch/touch-icon-196x196.png">

<!-- Tile icon for Win8 (144x144 + tile color) -->
<meta name="msapplication-TileImage" content="images/touch/ms-touch-icon-144x144-precompo
<meta name="msapplication-TileColor" content="#33720F">

<!-- Generic Icon -->
<link rel="shortcut icon" href="images/touch/touch-icon-57x57.png">
```

App Manifest

JD.COM 京东

使用 link 标签将 manifest.json 部署到 PWA 站点 HTML 页面的头部

```
1  <link rel="manifest" href="path-to-manifest/manifest.json">
```

包含信息

```
{
  "name": "京东摄影金像奖",
  "short_name": "京东摄影",
  "icons": [
    {
      "src": "//misc.360buyimg.com/channel/photography_awards/m/1.0.0/img/icons/192x192.png",
      "sizes": "192x192",
      "type": "image/png"
    }, {
      "src": "//misc.360buyimg.com/channel/photography_awards/m/1.0.0/img/icons/512x512.png",
      "sizes": "512x512",
      "type": "image/png"
    }
  ],
  "start_url": "/jdpa_m.html",
  "background_color": "#f3f3f4",
  "display": "standalone",
  "theme_color": "#e4393c"
}
```

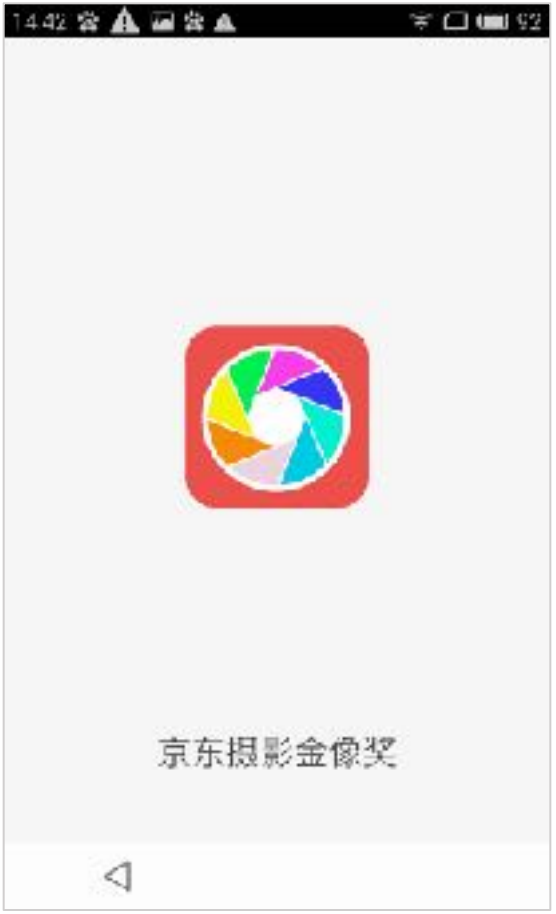
App Manifest

JD.COM 京东

安装横幅

主屏幕应用显示

启动画面



App Manifest

JD.COM 京东

- 改善应用体验

1. 添加启动画面：图标、背景色等替代白屏
2. 设置显示类型：全屏、导航栏和工具栏
3. 指定显示方向：“横屏”、“竖屏”
4. 设置主题色：控制浏览器 UI 的颜色
5. 原生应用安装横幅:推广原生应用

Notification&push API

JD.COM 京东

需要用户授权


```
function askPermission() {
    return new Promise(function (resolve, reject) {
        var permissionResult = Notification.requestPermission(function (result) {
            // 旧版本
            resolve(result);
        });
        if (permissionResult) {
            // 新版本
            permissionResult.then(resolve, reject);
        }
    })
    .then(function (permissionResult) {
        if (permissionResult !== 'granted') {
            // 用户未授权
        }
    });
}
```

App-shell

JD.COM 京东

App Shell 架构是构建 PWA 应用的一种方式，它通常提供了一个最基本的 Web App 框架，包括应用的头部、底部、菜单栏等结构

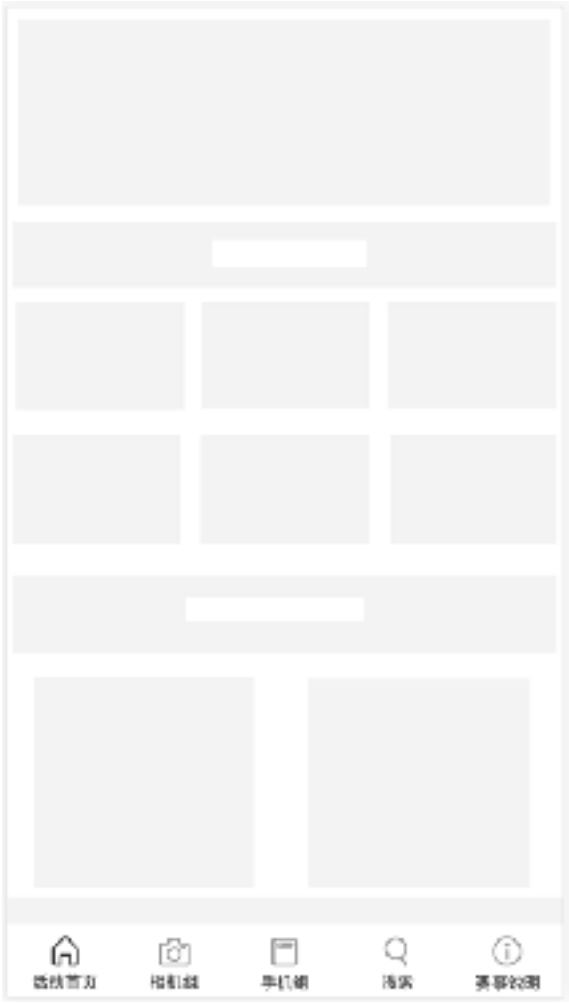
JD.COM 京东

Skeleton

1. Native APP 在实际内容加载出来之前会有一些 **占位图片**，其结构和实际加载内容相似；

2. 传统的 Web 站点加载前为白屏，前端能优化的仅是缩短这个白屏时间；

3. PWA 的缓存机制，我们现在已经有能力让 Skeleton 也出现在 Web App 上取代白屏



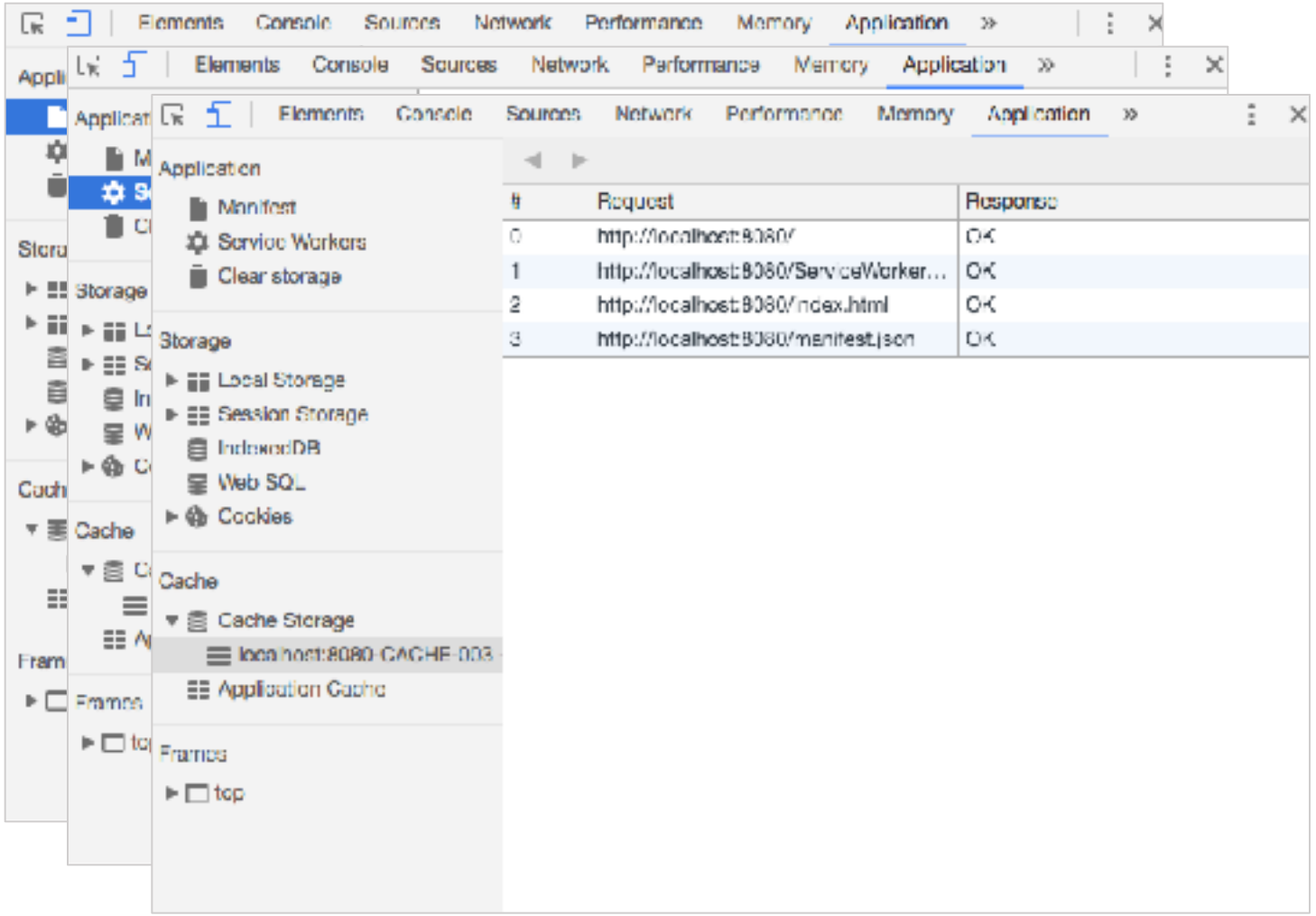
怎样做?

JD.COM 京东

How?

如何调试?

JD.COM 京东



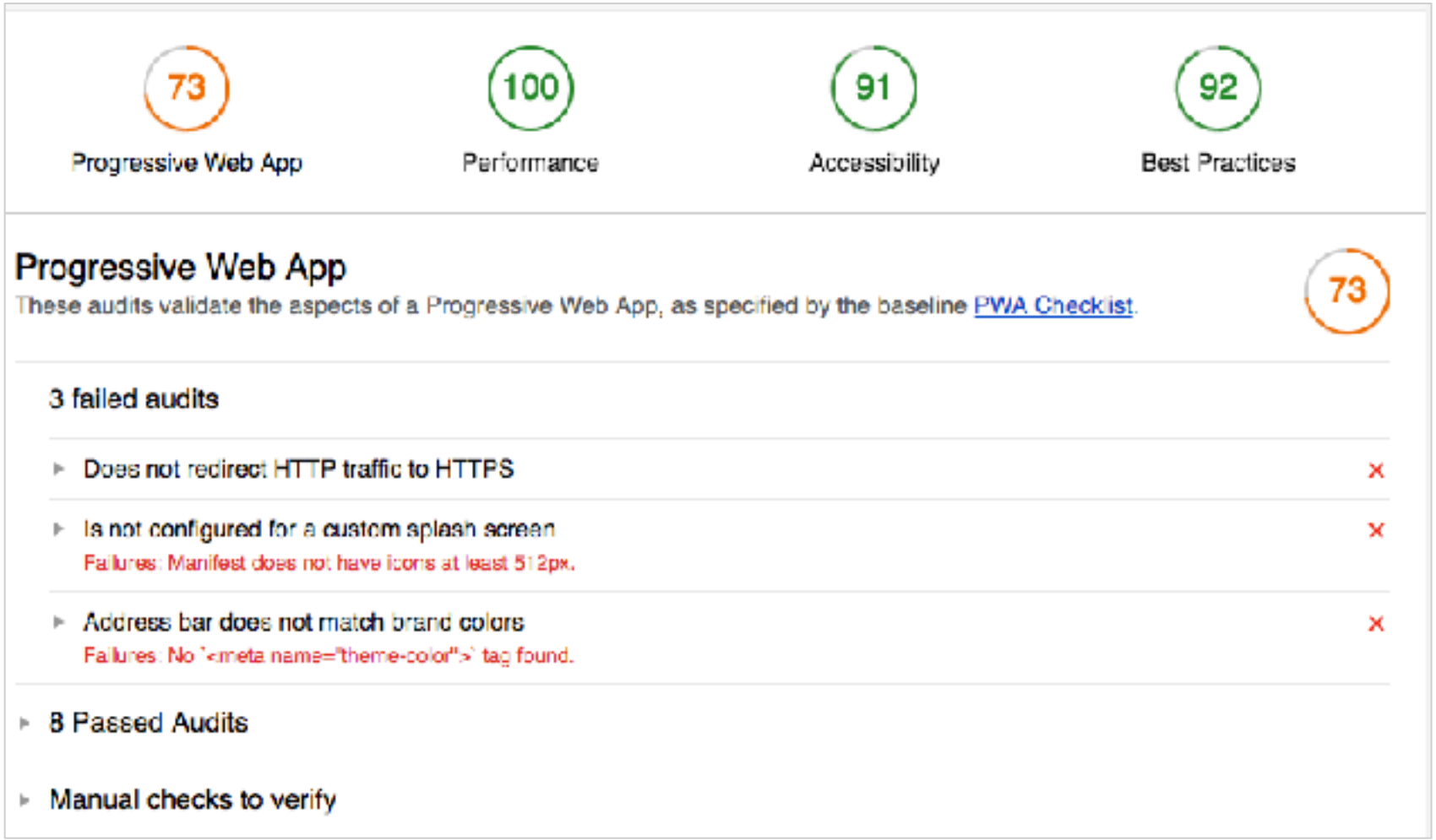
查看全部sw：chrome://inspect/#service-workers

查看sw详情：chrome://serviceworker-internals/

如何检查? -Lighthouse

JD.COM 京东

使用 Lighthouse 审查 WebApp



web-pwa

JD.COM 京东

web-pwa: https://www.npmjs.com/package/web-pwa

```
38  import SW,{Notify,WebCaches} from 'web-pwa';
39
40  window.onload = function(){
41      SW.register('sw.js');
42      var tableName = 'prefetch-cache-v1';
43      WebCaches.table(tableName).addRow('/app.js')
44      .then(res=>{
45          // res: 成功
46      })
47      Notify.request() // 请求推送权限
48      .then(permission=>{
49          // 用户同意
50          Notify.show('villianhr','Hello Pwa')
51          .onclick(event=>{
52              event.close(); // 关闭当前 Notification
53              Notify.focus(); // 聚焦窗口
54          })
55      })
56  }
```

1. SW: 主要处理主线程 JS Service Worker 的相关行为；例如：注册，发送消息等
2. WebCaches: 用来处理 CacheStorage 缓存的相关操作；
3. Notify: 根据 new Notification() 来完成主线程 JS 的消息推送；

sw-precache

JD.COM 京东

sw-precache 可以用来生成配置使 PWA 在安装时进行静态资源的缓存;

此模块可用于基于 JavaScript 的构建脚本中，例如使用 gulp 编写的脚本;

功能	摘要
预缓存 App Shell	网络应用的 Shell（它的核心 HTML、JavaScript 和 CSS）可以在用户访问页面时预缓存。
构建时集成	集成到您的现有构建流程中：Gulp、Grunt或 命令行。
实时更新	构建中的更改会更新服务工作线程脚本。用户可以获取更新，但您无需手动更新内容或缓存。
没有网络，也没关系	无论网络是否可用，都能通过缓存优先的方式快速提供您的静态资源。

sw-toolbox

JD.COM 京东

sw-toolbox 提供了动态缓存使用的通用策略, 这些动态的资源不合适用 sw-precache 预先缓存

功能	摘要
运行时缓存	在运行时或第一次使用时，可以缓存较大或不经常使用的资源，如图像。
离线回退	在线时从网络加载全新的图像、API 响应或其他动态内容，但离线时将回退到缓存占位符。
告别 Lie-Fi	在网络过慢时，通过自动回退到缓存响应应对 lie-fi。
应对缓存膨胀	无需再缓存上个月的图像，最近最少使用和基于时间的缓存到期时间有助于释放空间。

Webpack plusins

JD.COM 京东

https://github.com/goldhand/sw-precache-webpack-plugin

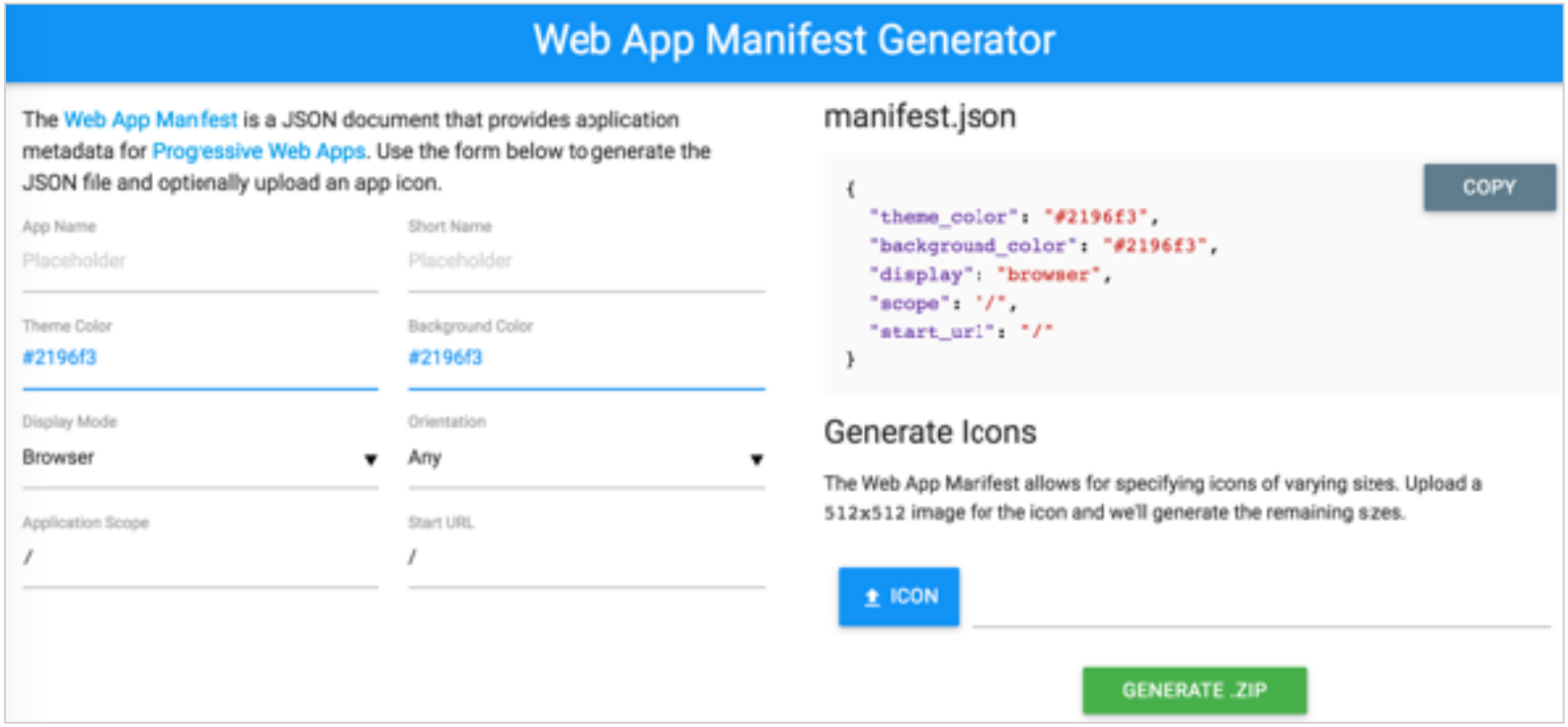
```
var path = require('path');
var SWPrecacheWebpackPlugin = require('sw-precache-webpack-plugin');

module.exports = {
  context: __dirname,
  entry: {
    main: path.resolve(__dirname, 'src/index'),
  },
  output: {
    path: path.resolve(__dirname, 'src/bundles/'),
    filename: '[name]-[hash].js',
  },
  plugins: [
    new SWPrecacheWebpackPlugin(
      {
        cacheId: 'my-project-name',
        filename: 'my-service-worker.js',
        maximumFileSizeToCacheInBytes: 4194304,
        minify: true,
        runtimeCaching: [{
          handler: 'cacheFirst',
          urlPattern: /[.]mp3$/,
        }],
      }
    )
  ]
}
```

Manifest Generator

JD.COM 京东

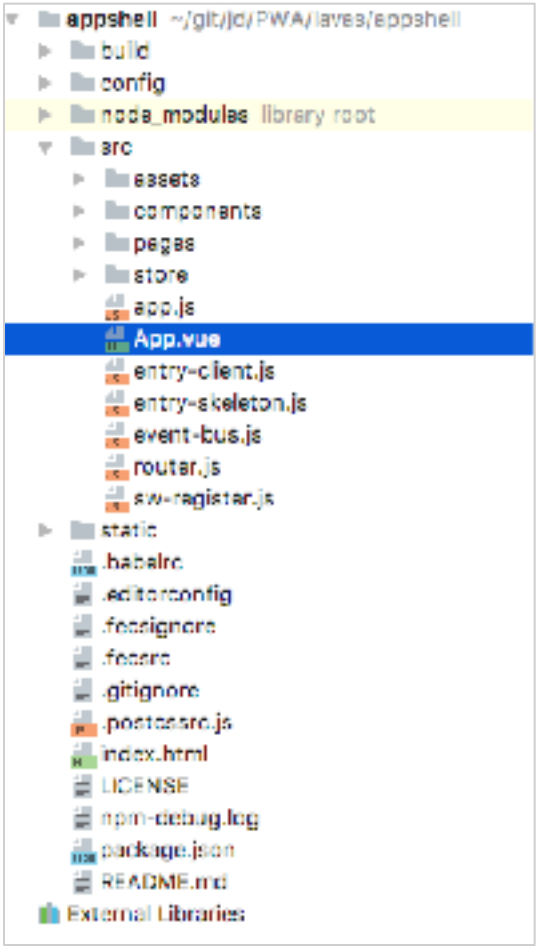
https://app-manifest.firebaseapp.com/



Lavas

JD.COM 京东

Lavas 是一个基于 Vue 的 PWA (Progressive Web Apps) 完整解决方案。将 PWA 的工程实践总结成多种 Lavas 应用框架模板，帮助开发者轻松搭建 PWA 站点，且无需过多的关注 PWA 开发本身。



哪里使用?

JD.COM 京东

Where?

使用场景-兼容性

JD.COM 京东

各项技术的支持度:

1. App Manifest： 59.15%
2. Service Worker： 64.19%+9.47%=73.66%
3. Notifications API： 35.75%+5.42% =41.17%
4. Push API： 71.68%+1.61%=73.29%
5. Background Sync： Chrome 49 以上均支持

Can I use 统计

使用场景-兼容性

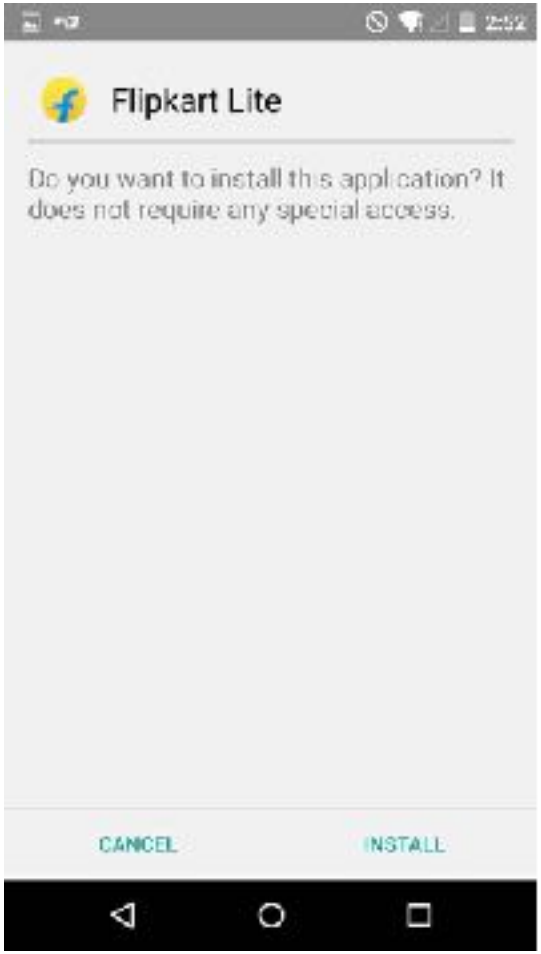
JD.COM 京东

标准的支持度:

1. 各项基本技术，都已经或正在被标准化；如W3C、ECMA;
2. 各浏览器厂商背后助力;
3. 是web 社区与整个 web 规范;

使用场景-未来计划

JD.COM 京东



PWA安装

App列表

系统设置

使用场景

JD.COM 京东

PWA有必要吗？京东有适用场景吗？

1. 兼容性不好：渐进式的，浏览器支持逐步完善；
2. 非实时数据：假设用户离线状态是临时的，应对网络不稳定；
3. 离线状态操作：保证浏览，合理保存用户操作，及时同步；
4. 可以参考：flipkart、饿了么；

Challenge！

PWA改造成本大吗?

第一步：应该是安全，将全站 HTTPS 化，因为这是 PWA 的基础，没有 HTTPS，就没有 Service Worker；

第二步：应该是 Service Worker 来提升基础性能，离线提供静态文件，把用户首屏体验提升上来；

第三步，App Manifest，这一步可以和第二步同时进行

后续，再考虑其他的特性，离线消息推送等

前端的挑战

JD.COM 京东

需要做什么?

- 不是前端技术推进，需要后端达成共识，产品认可;
- 深入到产品设计、交互及UI设计中，offline-first宗旨;
- 前端任重道远，直接关系用户体验;

THANK YOU!