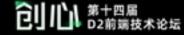


2019-12-14 杭州和达希尔顿逸林酒店





微前端的沙盒的技术体系

- ① 沙盒应该做什么
- ②沙盒像什么
- ③ 沙盒怎么做
- ④ 字节跳动的沙盒做什么
- ⑤沙盒模式中埋点、系统采样的设计





① 沙盒应该做什么

- · 古老的 Iframe
 - 一个站点页面拆成 N 个 frame
 - · 每个 frame 单独一个独立域名
 - 独立上下线
 - 独立运行时





① 沙盒应该做什么

- 古老的困难
 - deeplinking
 - 共用数据:登录身份、站内信、跨模块通信
 - 共用代码、加载优化、运行优化





② 沙盒像什么

- 1 Docker
 - 开发者必须体会不到环境区别
 - 运行时没有环境差异
 - 服务端微服务的基石





②沙盒像什么

- 2. Docker 时代之前的微服务
 - 虚拟机使用复杂,维护成本巨大
 - 资源消耗
 - 镜像启动
 - 进程通信
 - 直到 Docker 普及
 - 前端的"微服务"在浏览器环境下并没有



门门 第十四届 D2前端技术论坛

② 沙盒像什么

- 3. 微前端的情况
 - 前端沙盒像浏览器 Docker
 - Iframe 像虚拟机
 - 字节跳动使用前端沙盒





③ 沙盒怎么做

- 1. 参考<u>单核、操作系统进程</u>模拟进程切换策略
 - JavaScript 是单线程的
 - 通过对路由切换的封装,模拟单进程
 - 通过对事件循环封装,模拟单核多进程





③ 沙盒怎么做

- 2. 用 Context 切换模拟线程安全
 - 新沙盒即将激活时,查找当前激活中的沙盒
 - 保存现场,存储 context
 - 恢复之前的 context





③ 沙盒怎么做

- 3. Context 切换的笛卡尔积
 - 比较并切换
 - 沙盒数量 N 的笛卡尔平方
 - 退回"初始" context
 - 恢复之前 diff 的 context





- 1. CSS 的干扰
 - 独立开发、混合加载
 - HTML 标准的 CSS 作用域
 - Scoped CSS
 - Shadow DOM
 - CSS module, CSS in JS

•





- 1. CSS 的干扰

 - CSS module, CSS in JS
 - DOM header





- 1. CSS 的干扰
 - •
 - 单核多进程的沙盒情况
 - •





- 1. CSS 的干扰
 - •
 - CSSStyleSheet.cssRules
 - •





- 2. 全局变量的干扰
 - Polyfill 等差异巨大
 - 例如 generatorRuntime
 - 组件模块化
 - 全局的外部环境

•





- 2. 全局变量的干扰
 - •
 - Identifier
 - let
 - const
 - class
 - •





- 2. 全局变量的干扰
 - configable
 - location





- 3.所有进程安全的对象
 - · DOM 沙盒等
 - cookie
 - localstorage





⑤沙盒下的埋点、统计等

- 1. 埋点数据的缓存创建
 - 全局数据(uid 等)默认缓存本地
 - 缓存跟随沙盒切换
 - 两级缓存:沙箱内全局、系统全局





⑤字节跳动沙盒下的埋点、统计等

- 2. 埋点数据的发送
 - 异步发送
 - 触发时机在沙盒外、缓存跟随沙盒切换
 - 全局缓存和本地缓存统一本地存储





⑤字节跳动沙盒下的埋点、统计等

- 3. Console 回收
 - 干净体面的正规军
 - 控制 sourcemap
 - 向 log 中注入 callstack
 - 额外加入快照





⑤字节跳动沙盒下的埋点、统计等

- 4. sourceMapping
 - 原理:文本的映射
 - new Function
 - 管理发布平台



.





调研问卷

