

深入浅出前端沙箱技术

前言

今天主要从浏览器层面到 SPA 微前端层面讲解一下隔离的概念和技术。

浏览器

浏览器多进程架构

沙箱隔离 & 站点隔离

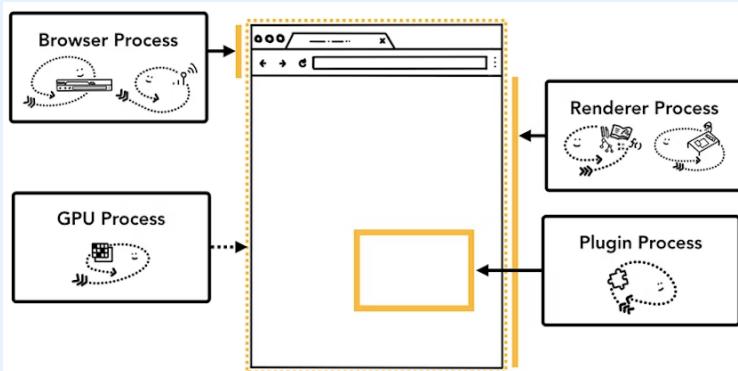
V8 隔离

SPA 微前端

JS 隔离

DOM 隔离

浏览器多进程架构



浏览器多进程架构

Browser 主进程：处理网络请求、用户输入输出、地址栏 URL 管理、书签管理、回退与前进按钮、文件访问、Cookie 数据存储等。

Renderer 进程：负责标签页和 iframe 所在 Web 应用的页面渲染和 JavaScript 执行。

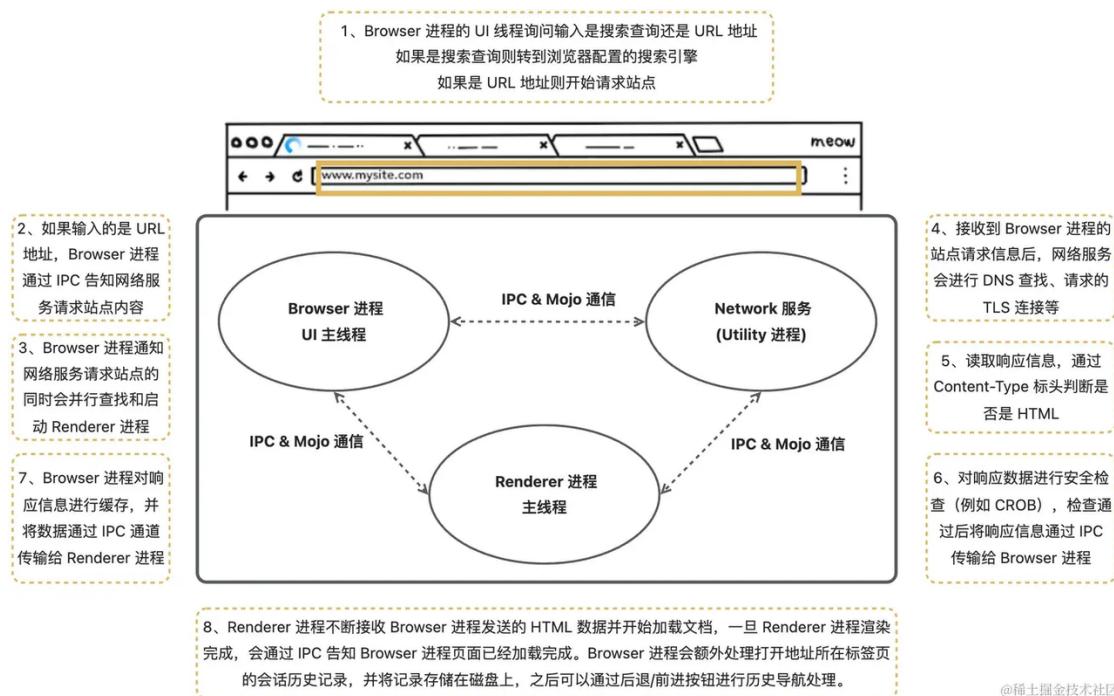
？新开的标签页和 Renderer 进程是 1: 1 的关系吗？

Chrome 浏览器的多进程架构是可伸缩的架构系统吗？例如在 Android 中是如何处理的？

浏览器多进程架构的好处是什么？

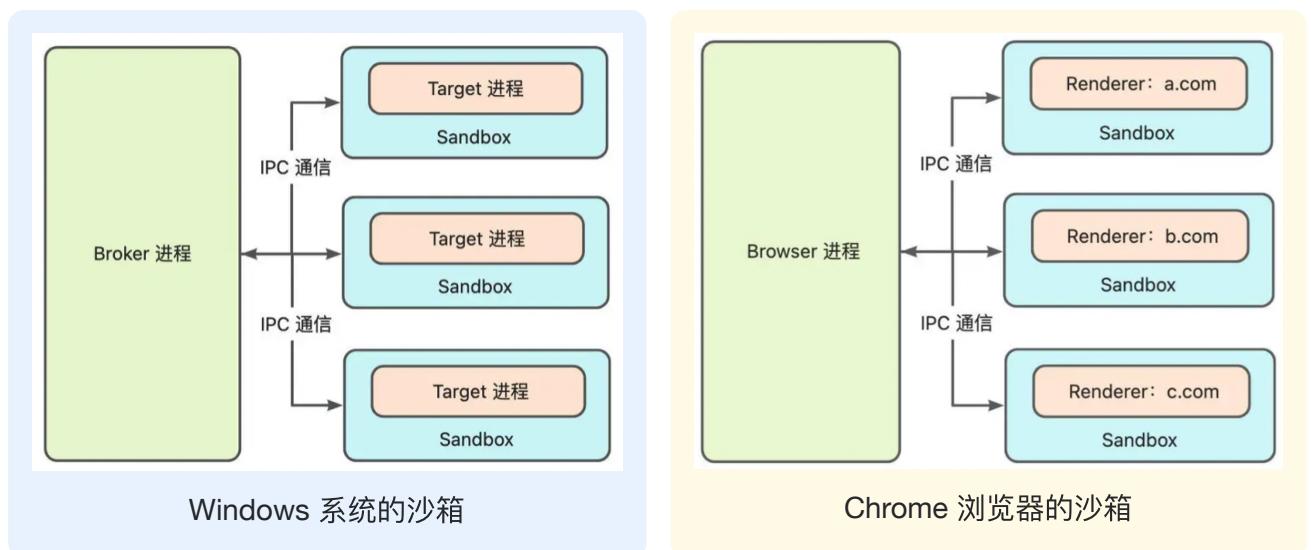
多进程协作

? 客户端在定制的 Chromium 源码 (CEF 容器) 里可以做哪些性能优化处理?



@稀土掘金技术社区

浏览器沙箱



? Chrome 浏览器的插件进程是否已经沙箱化?

例如从插件进程和 Renderer 进程出发, 可以感知哪些沙箱策略的差异性?

Broker 进程: 派生 Target 进程、管理 Target 进程的沙箱策略、代理 Target 进程执行策略允许的操作。

Target 进程: 运行时受到沙箱策略的管控，会通过 IPC 告知 Broker 进程代理执行操作，例如网络请求、文件访问（磁盘）、用户输入输出的访问（设备）。

沙箱的作用

隔离环境

三方应用运行在隔离的环境中，防止恶意攻击和干扰

权限管控

文件系统权限、网络权限、输入输出设备权限

安全隐私

保护用户的安全隐私、防止用户数据被恶意访问和篡改

通信机制

进程和应用间通信

浏览器同源策略安全风险

? 如果多个 Web 应用共享同一个 Renderer 进程，浏览器如何进行 Web 应用隔离？

不同源的标签页应用和内部的 iframe 应用会处于同一个 Renderer 进程，可能会产生哪些安全风险？

获取 Web 应用的 Cookie 和 HTML 5 存储数据

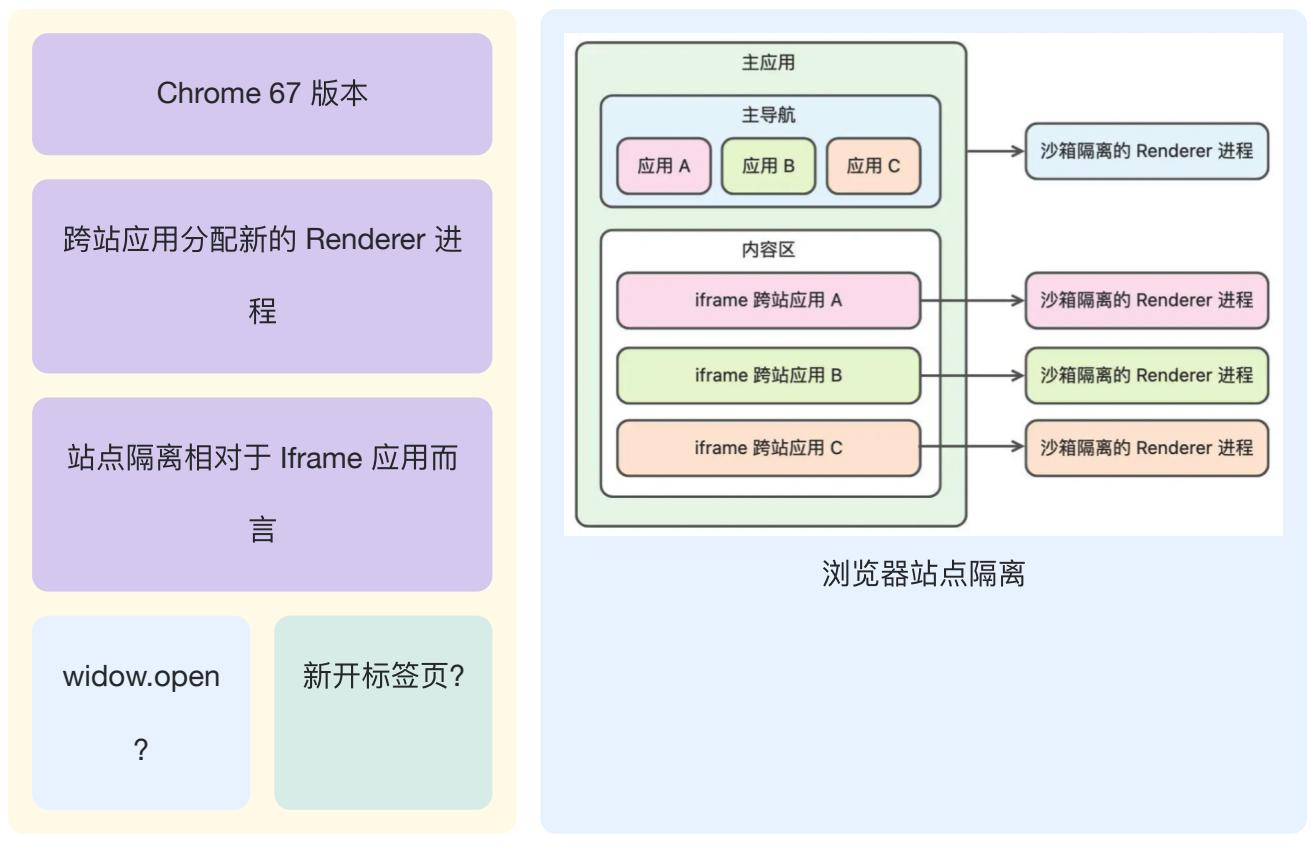
获取 Web 应用的 HTML、XML 和 JSON 数据

绕过 [X-Frame-Options](#) 加载 iframe 应用（例如钉钉的页面被 iframe 嵌套）

获取浏览器保存的密码数据

共享 Web 应用的授权权限，例如地理位置

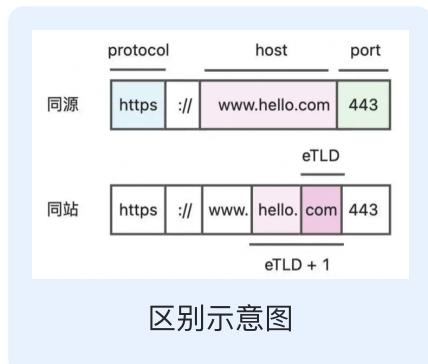
浏览器站点隔离



？ Chrome 浏览器为什么不实现源隔离？

window.open、Iframe 和新开标签页对于创建 Renderer 进程的策略有什么不一样吗？

同站和同源的区别



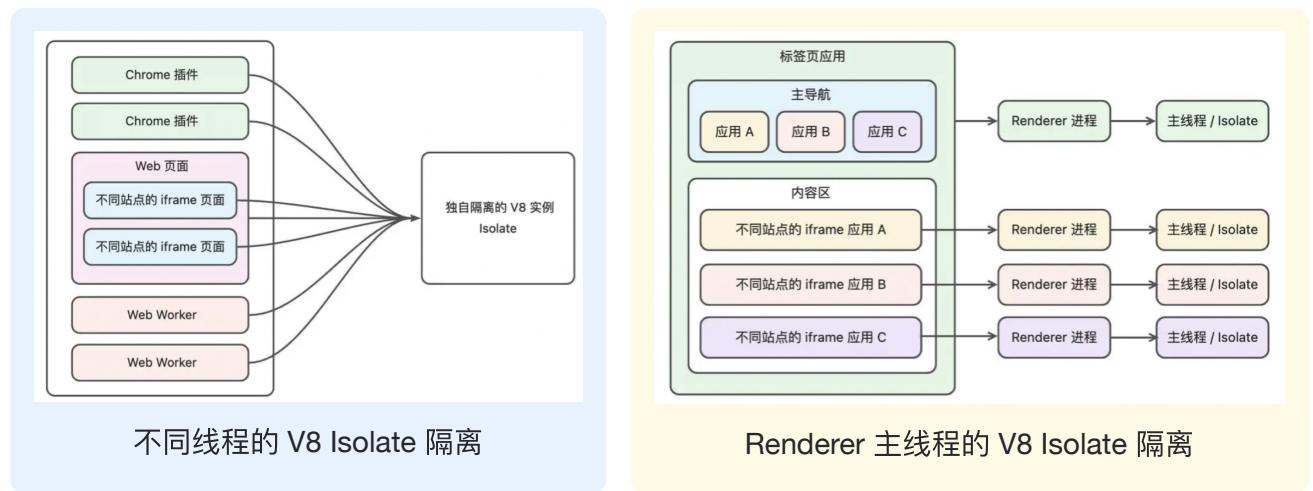
同源：协议（Protocol）、主机名（Host）和端口（Port）相同，则为同源。

同站：有效顶级域名（Effective Top-Level-Domain, eTLD）和二级域名相同，则为同站。

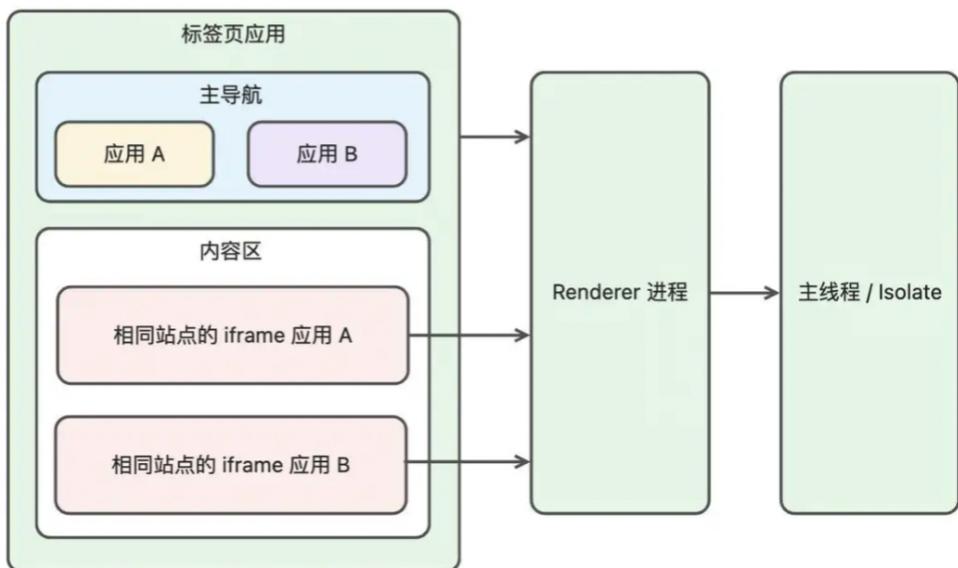
类型	应用 A	应用 B	判断结果
同源判断	https://www.hello.com:443	http://www.hello.com:443	跨域: 协议 (protocol) 不同
		https://www.world.com:443	跨域: 主机名 (host) 不同
		https://www.hello.com:80	跨域: 端口 (port) 不同
		https://www.hello.com/a.html:443	同源: 协议、主机名、端口相同
同站判断	https://www.hello.com:443	http://www.hello.com:443	同站: eTLD + 1 相同
		https://www.world.com:443	跨站: 二级域名不同
		https://www.hello.com:80	同站: eTLD + 1 相同
		https://www.hello.com/a.html:443	同站: eTLD + 1 相同

@稀土掘金技术社区

V8 隔离

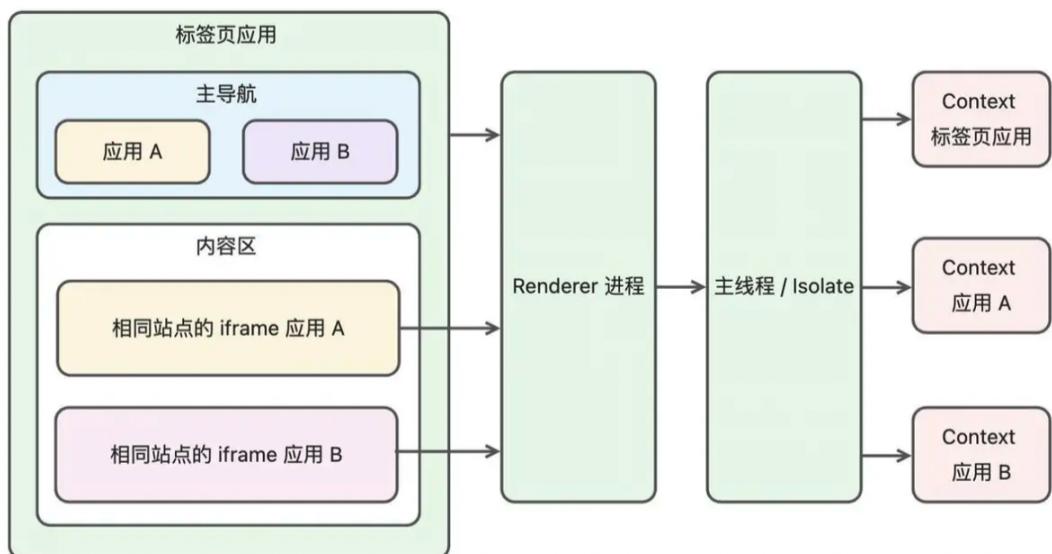


? Renderer 进程中的主线程和 V8 的 Isolate 隔离实例是 1:1 的关系吗?



相同站点应用共享 V8 的 Isolate 实例

? 同一个 Renderer 进程中不同 Web 应用的 V8 如何实现运行时的全局执行上下文 window 隔离?



V8 中的 Context (全局执行上下文)

Isolate: 在安全上用于物理空间的隔离，可以防止跨站攻击，有自己的堆内存和垃圾回收器等资源，不同的 Isolate 之间的内存和资源相互隔离，它们之间无法共享数据，是非常安全可靠的隔离。

Context: 有自己的全局变量、函数和对象等，默认情况下不同 Context 对应的 JavaScript 全局上下文无法访问其它全局上下文。

隔离概述

我们可以总结一下浏览器的隔离能力

新开标签页：Renderer 进程隔离

不同站点应用：Renderer 进程隔离 + V8 Isolate 隔离

同站应用：浏览上下文隔离 + V8 Context 隔离

SPA 微前端：？

标准能力

抱歉，浏览器没有出 JS 和 DOM 隔离相关的 Web API，我们只能间接查看其它能力

WebAssembly

Iframe

Web Worker

Shadow DOM

proposal-shadowrealm (提案)

❓ 上述能力中哪些适合做 SPA 应用的隔离技术？

隔离方案

如果 SPA 应用想要实现第三方应用的环境隔离，那么有如下几种方案选择：

DOM 隔离

Shadow DOM

iframe

Mock 隔离（非完整的隔离能力）

JS 隔离

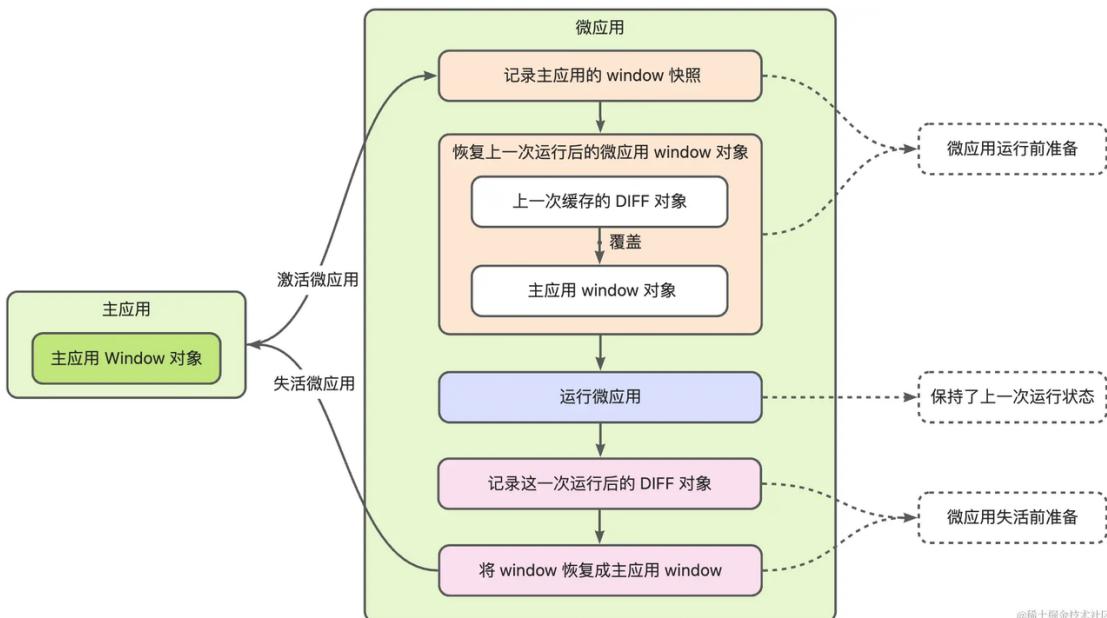
iframe

Mock 隔离（非完整的隔离能力）

社区方案

框架	JS 隔离	DOM 隔离（CSS 隔离）
qiankun	Mock 隔离	Mock 隔离 + Shadow DOM
wujie	iframe 隔离	Shadow DOM
MicroApp	Mock 隔离	Mock 隔离 + Shadow DOM
icestark	Mock 隔离	Mock 隔离 + Shadow DOM

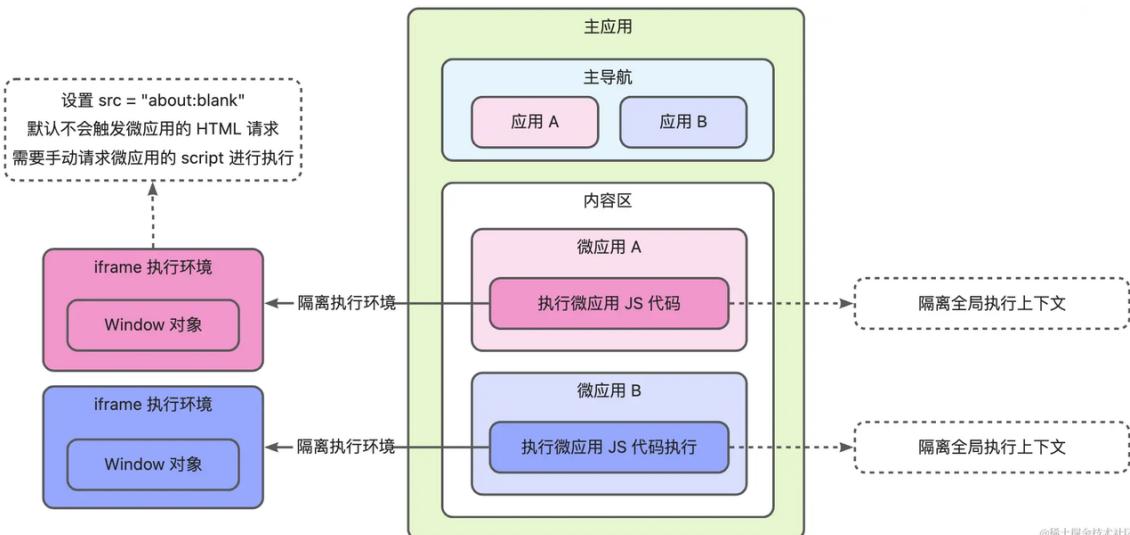
JS 隔离



@稀土掘金技术社区

快照隔离

? 快照隔离有什么限制吗？

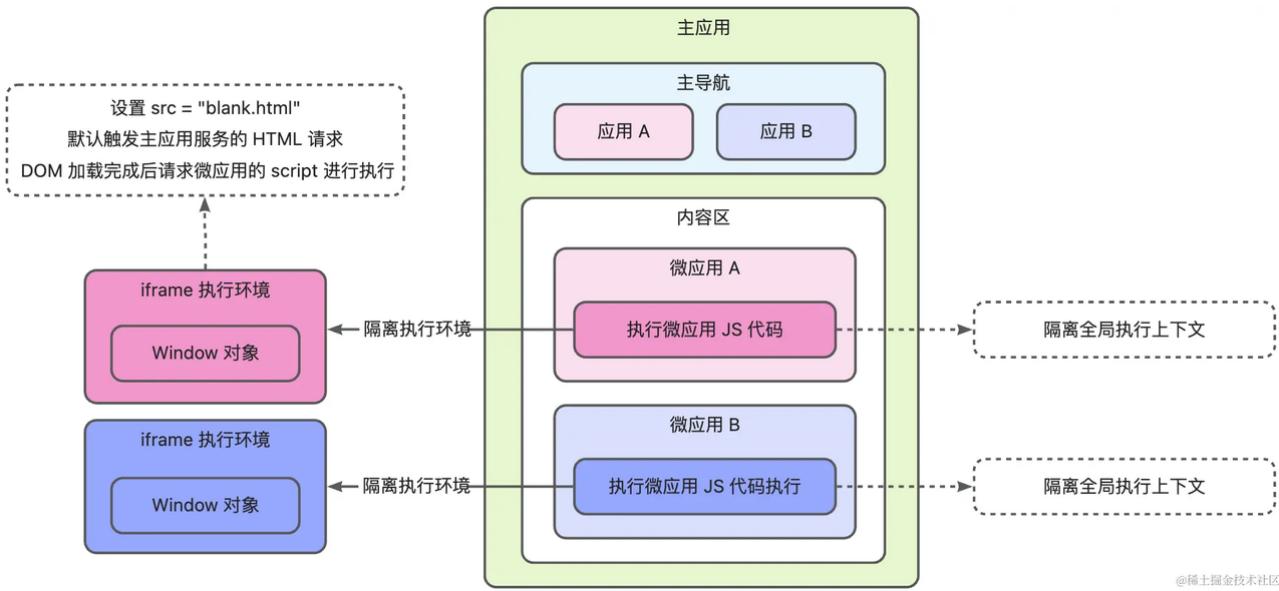


@稀土掘金技术社区

空白的 iframe 隔离

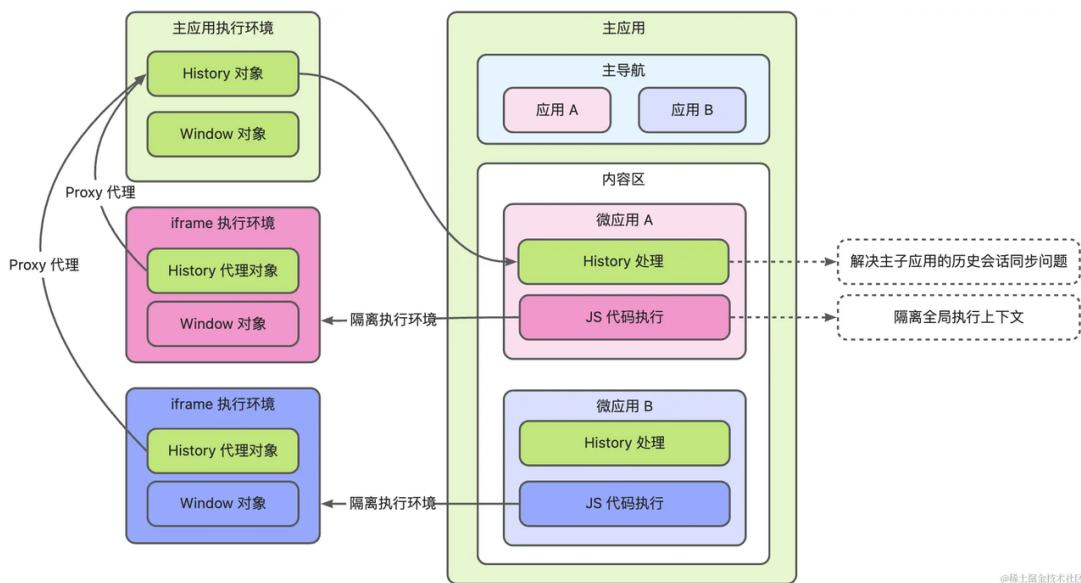
? 如何解决 `src=about:blank` 后的 History API 报错问题？

`src=about:blank` 后 Iframe 应用中发送的 Ajax 请求会存在跨域问题吗？



同源的 iframe 隔离

？该方案的缺点是什么？



空白的 iframe + Proxy 隔离 (逃生窗口)

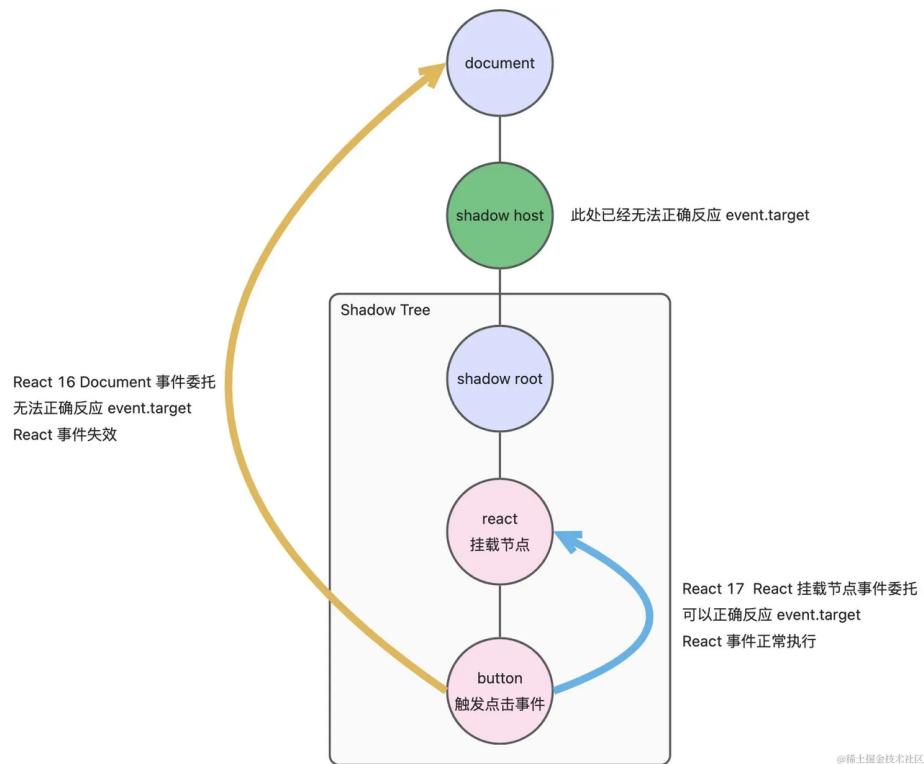
？如何解决 Modal 框相对于主应用居中的问题？因为 Iframe 和主应用浏览上下文隔离了

如何解决 Iframe 中的 URL 相对于主应用同步的问题？

社区是如何解决 Modal 框相对主应用居中的问题？比如要实现全屏的遮幕能力？

DOM 隔离

完整的 DOM 隔离只有 Shadow DOM 能力，该能力不仅可以隔绝 DOM 和样式，还可以隔绝事件



总结

更多细节可以查看 [《深入浅出微前端》小册](#)。