



11月12日 | 第74届前端早早聊

前端工程化 工程演进 业务进阶

SSR

依赖管理

容器化

工程实践

架构演进



10:00 SSR:《大型 Web 应用的性能提升之路》

王瑜

字节跳动
飞书文档
前端工程师

11:00 依赖:《NPM 依赖管理的复杂性》

范文杰

字节跳动
飞书 aPaaS 团队
资深前端工程师

12:00 实践:《得物前端研发工程链路实践》

庄再苦

得物
前端平台
前端工程师

2023 全年行程

2/4 健康指南
2/25 前端搞规划
3/18 前端搞插件
4/1 杭州 GPT 沙龙
4/8 Node.js
4/15 低代码无代码
5/7 前端搞管理
5/20 上海 GPT 沙龙
6/3 求职就业新起点
6/10 武汉 GPT 沙龙
6/17 前端性能优化
7/15 AI Chatbot
8/12 Night Of AI
9/2 前端搞构建
9/23 前端搞 AI
10/28 AI 线下沙龙
10/29 前端跨端方案
11/4 AI 商业与赛道
11/12 前端工程化
11/19 前端搞 Remote
11/25 玩转 Prompt
12/2 前端搞微前端
12/9 表格与表单
12/16 前端搞浏览器
12/23 前端质量保障


大型 Web 应用的性能提升之路

—— 聚焦服务端侧优化

字节跳动 王瑜

通过分享，可以收获

- 1.性能优化的系统性思路
- 2.服务端优化的收益和常见手段（侧重点）
- 3.页面渲染服务定位的思考（一起交流哈）



目录

1

前言

2

治理思路

3

服务端侧提升详解

4

结尾&QA

前言 —— 术语解释

- TTFB (Time To First Byte) : 是指从用户发起请求到浏览器从服务器接收第一个字节数据的时间。
- TTV (Time To Valid) : 是指从用户发起请求到浏览器确认页面有效并开始渲染的时间。
- TTI (Time To Interactive) : 是指从用户发起请求到页面变得可交互的时间。
- Page Server: 页面渲染服务, 用于生成静态或动态网页内容, 并将其发送到客户端浏览器。
- BFF (Backend For Frontend) : BFF 是一种架构模式, 其中后端服务器专门为前端应用程序提供数据和 API。它可以提高前端应用程序的性能和可维护性。
- SSR (Server-Side Rendering) : SSR 是一种在服务器端生成 HTML 内容的技术。它可以提高页面的加载速度, 因为服务器可以在请求时生成 HTML 内容, 而不是在客户端浏览器上进行渲染。
- 流式渲染: 在数据传输过程中逐步显示内容, 而不是等待所有内容都传输完毕后再显示。这种技术可以提高页面的加载速度和用户体验

前言

8秒原则："等待网页加载8秒之后，用户将失去耐心，从而停止浏览该网页"

对于成熟的前端应用而言，性能是仅次于“可用性”的核心指标，复杂应用更甚，比如编辑器、表格，不做优化的话，打开时间都在数秒甚至数十秒。

前言——“大型”Web 应用

1. 产品(业务)复杂度： 多形态、多功能融合等

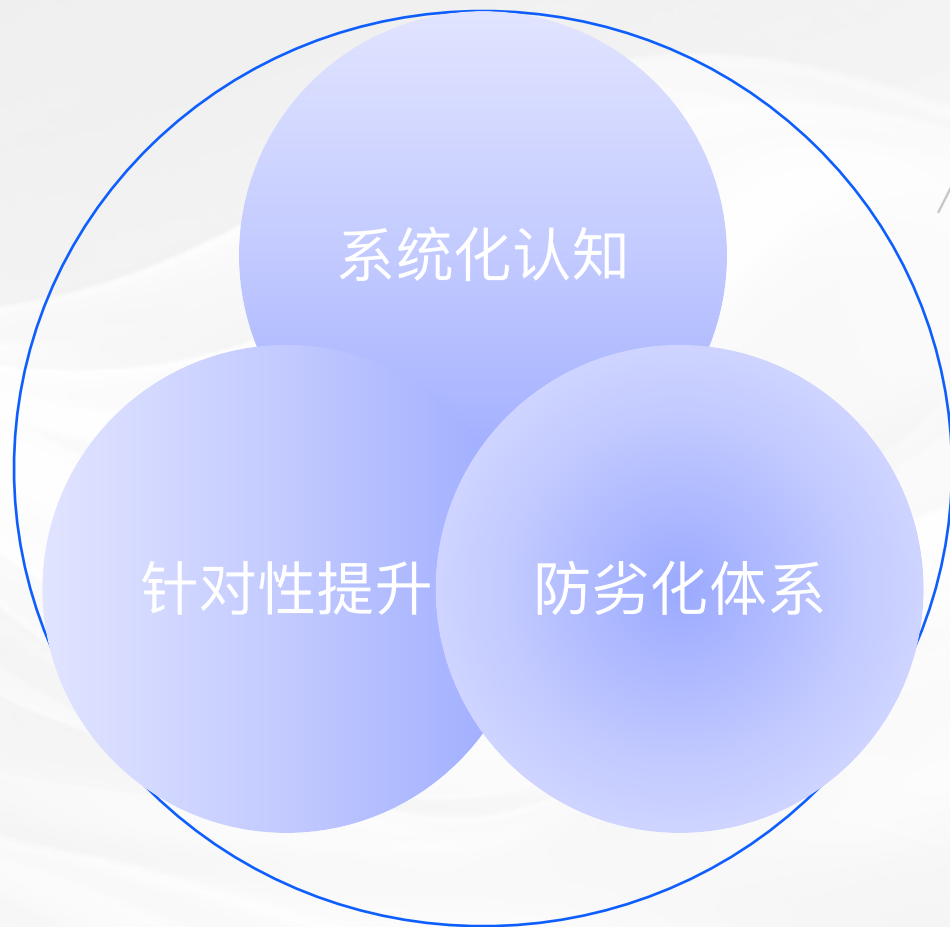
2. 技术复杂度： 技术实现复杂、规模化、历史包袱等

前言——“大型”Web 应用

1.没有银弹

2. 治理需要体系化

解决思路



系统性认知

浏览器打开页面的关键步骤



DNS 解析



TCP 建连



请求HTML 并返回



构建 DOM 树



请求 js、css资源

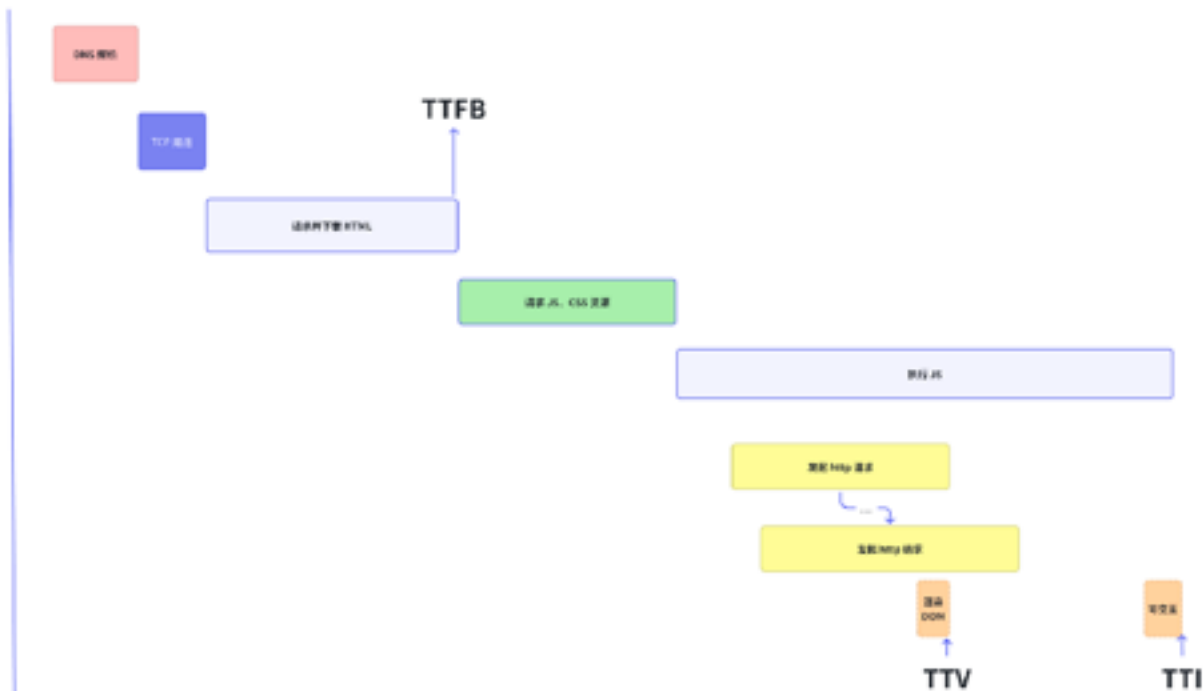


执行 JS，发起其他请求



渲染页面树

系统性认知



系统性认知 —— DNS 解析

- 减少域名使用

系统性认知 —— TCP 建连

HTTP 2.0: 多路复用、头部压缩

系统性认知 —— 请求 HTML 并返回

- BFF
- 数据注入
- SSR
- 流式渲染

系统性认知 —— 构建 DOM 树

- 减少 DOM 树数量
- 懒加载和分页加载

系统性认知 —— 请求 JS、css 资源

- 尽可能使用 cdn
- 资源包大小平均
- dead code 移除
- repeat code 移除

系统性认知 —— 执行 JS，发起 http 请求

- 非首屏 JS 延后加载执行
- 减少首屏 http 请求
- 优化请求时延

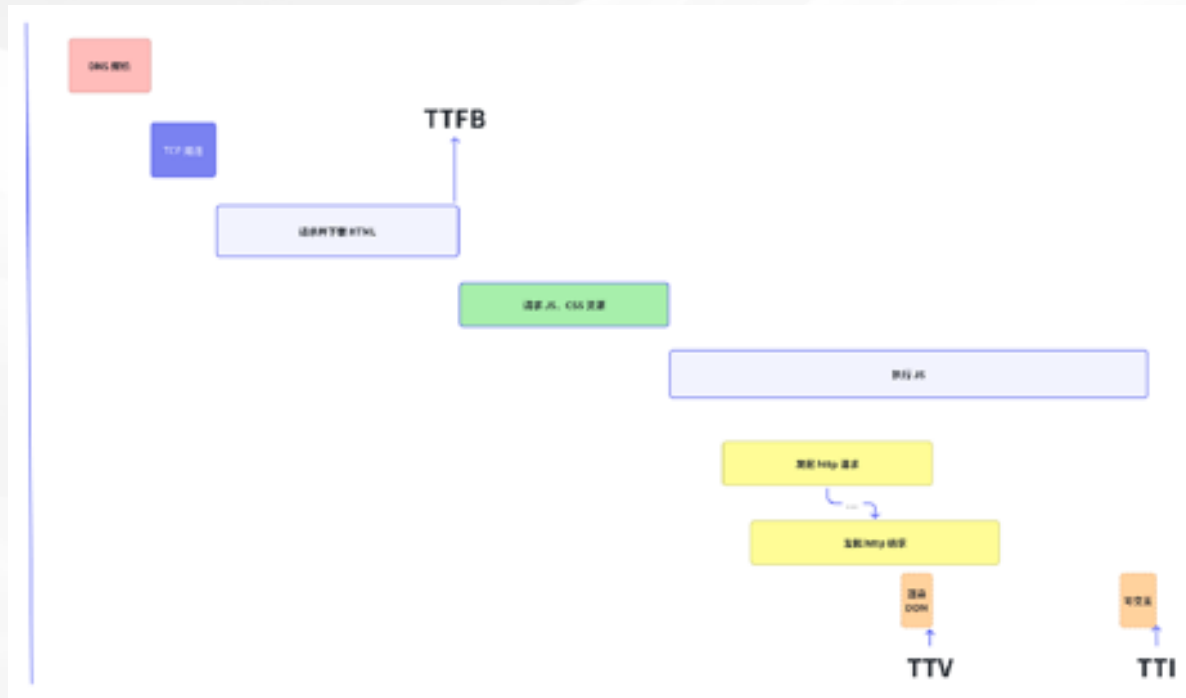
系统性认知 —— 渲染页面树

- 减少页面重绘

服务端侧提升详解

服务端侧提升

完全未经服务端优化的表现



服务端侧提升——接口聚合

作用：减少浏览器请求数

成本：需要接口聚合服务 or 改造前端获取数据方式

服务端侧提升——接口聚合

1. 简易的胶水层 —— 基于微服务的 BFF
2. 网关 接口编排 —— 基于 API 网关的 BFF
3. GraphQL —— 基于 serverless 的 BFF

服务端侧提升——数据注入

作用：

- 部分请求浏览器完全无需发送请求
- 排除用户侧网络干扰：弱网、低端机
- 内网调用性能更好

成本：需要页面渲染服务

服务端侧提升——数据注入

业内主流框架是怎么做的? —— Next.js

- `getInitialProps`
- `getServerSideProps`
- `getStaticProps`



服务端侧提升——数据注入

业内主流框架是怎么做的？—— Remix

Loader



```
1  import { useLoaderData } from "@remix-run/react";
2
3  import { prisma } from "../db";
4
5  export async function loader() {
6    return json(await prisma.user.findMany());
7  }
8
9  export default function Users() {
10   const data = useLoaderData<typeof loader>();
11   return (
12     <ul>
13       {data.map((user) => (
14         <li key={user.id}>{user.name}</li>
15       ))}
16     </ul>
17   );
18 }
```

服务端侧提升——数据注入

不借助框架该如何实现？

模板 + 数据 = HTML

服务端侧提升——数据注入

常见的模板引擎

- Nunjucks
- ejs
- Pug (Jade)
- ...

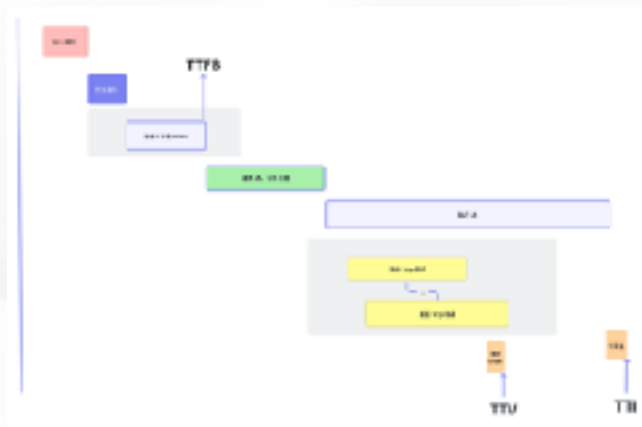
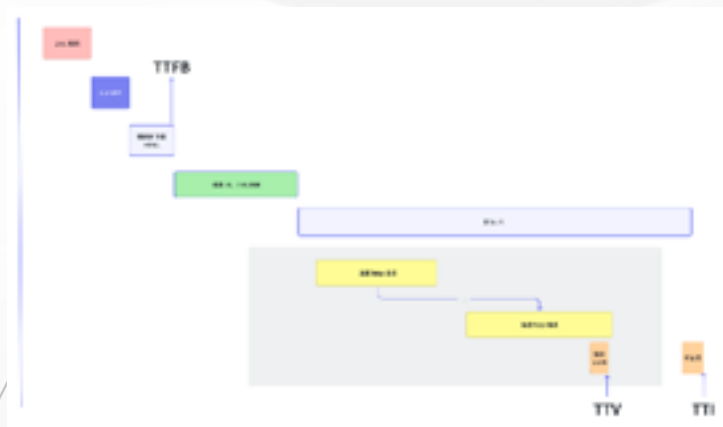
服务端侧提升——数据注入

服务端获取数据的方式

- http 请求
- rpc 请求（更快、更稳定）

服务端侧提升——数据注入

优化效果



服务端侧提升——SSR

实现方式

- 无头浏览器
- Next.js 等同构框架
- JSDom + React 服务端渲染方法

服务端侧提升——SSR

无头浏览器

实现思路	使用 puppeteer 加载页面，得到期望的 Dom 结构，返回给前端
优势	无需处理前端代码在服务端渲染问题，相对简单
劣势	<ul style="list-style-type: none">• 仅支持 Dom，不支持 canvas• 资源消耗多
适合	无 canvas 内容的前端页面，渲染量整体较小的应用

服务端侧提升——SSR

同构框架

实现思路	框架层面支持编译出 兼容浏览器、Node 两个运行时的产物
优势	DX 体感较好，相对 Mock，心智负担更低
劣势	前端框架前置需要支持同构编译
适合	新项目选型，可支持使用同构框架

服务端侧提升——SSR



Next.js

核心思想：

1. SSG、SSR、CSR
2. 开发、部署配置化、约定式

服务端侧提升——SSR



Remix

核心思想：

1. 拥抱 Web 原生标准
2. 嵌套路由 – prefetch
3. 完全由服务端下发数据 —— 聚焦于 loader

服务端侧提升——SSR



核心思想：html 中返回了首屏可交互的所有内联 JS，FCP 约等于 TTI

服务端侧提升——SSR



1. 序列化交互事件
2. 将所有交互事件冒泡，处理方法也以内联脚本形式返回
3. 非首屏必需脚本，切割、延迟加载

服务端侧提升——SSR



限制：

非基于 React 体系，无法享受 react 升级带来的福利，比如 streaming

服务端侧提升——SSR

JSDom + React Render

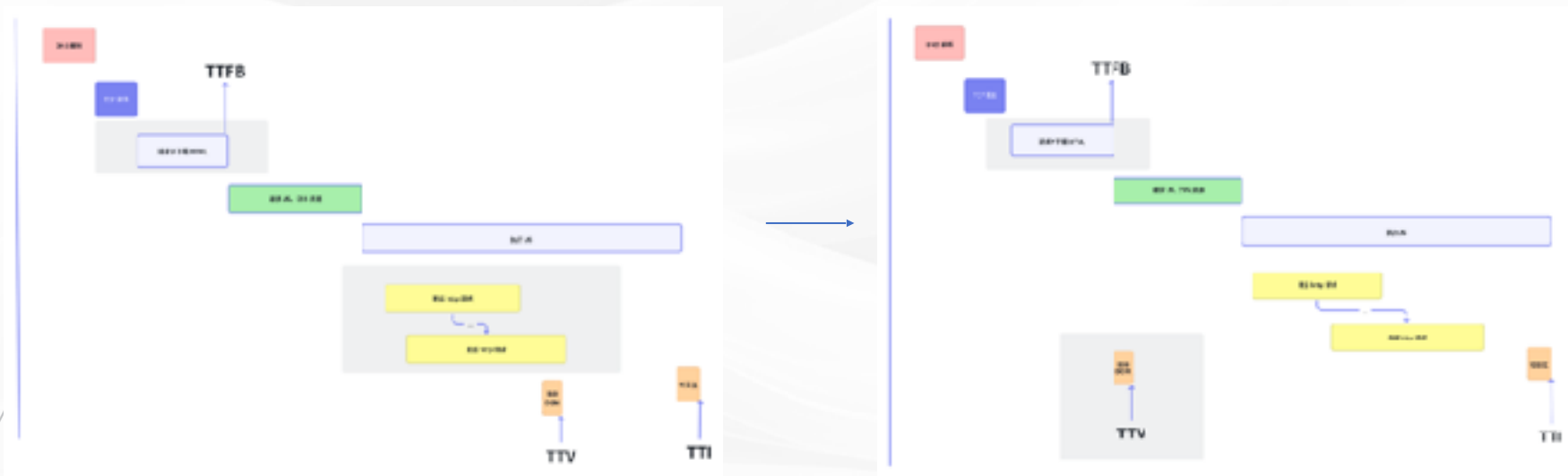
实现思路	通过 Mock 浏览器的 执行环境，加上 react-dom 官方支持的服务端方法在服务端渲染出 SSR Dom
优势	无需项目改造，实现成本低
劣势	<ul style="list-style-type: none">• 需要mock执行环境，存在编码认知差异• 前端需要编译 csr + ssr 两套 js 资源
适合	大型 React 项目快速支持 ssr，存量逻辑不支持同构且改造量大

服务端侧提升——SSR

没有万能的框架，因地制宜，适合业务的才是最好的

服务端侧提升——SSR

优化效果



服务端侧提升——SSR

渲染耗时长怎么办？

- 优化渲染
- 增加缓存

服务端侧提升——SSR

缓存优化

几百 ms



几十 ms

服务端侧提升——SSR

缓存陷阱 —— 清除机制

1. 设置缓存有效期
2. 明确失效规则
3. 尽可能提高命中率（首屏数据、剪枝）

服务端侧提升——SSR

缓存陷阱 —— 击穿

当热门命中的 key 失效后，导致下游流量激增，引发下游过载

服务端侧提升——SSR

缓存陷阱 —— 击穿

- 1.本地锁
- 2.分布式锁
 - 1.redis 社区实现 redlock
 - 2.redis lua 脚本原子操作

服务端侧提升——SSR

缓存陷阱 —— 穿透

指未命中的 key，流量都打到下游服务

服务端侧提升——SSR

缓存陷阱 —— 穿透

- 1.布隆过滤器
- 2.空值设置
- 3.前置拦截恶意无效请求，比如 WAF

服务端侧提升——SSR

缓存陷阱 —— 热 Key

副作用：

- 1.是引发缓存击穿的一个潜在条件（qps 很高的页面，突然缓存过期，就有可能引发击穿）
- 2.是持续访问 Redis，Redis 的 IO 负载也会增加

服务端侧提升——SSR

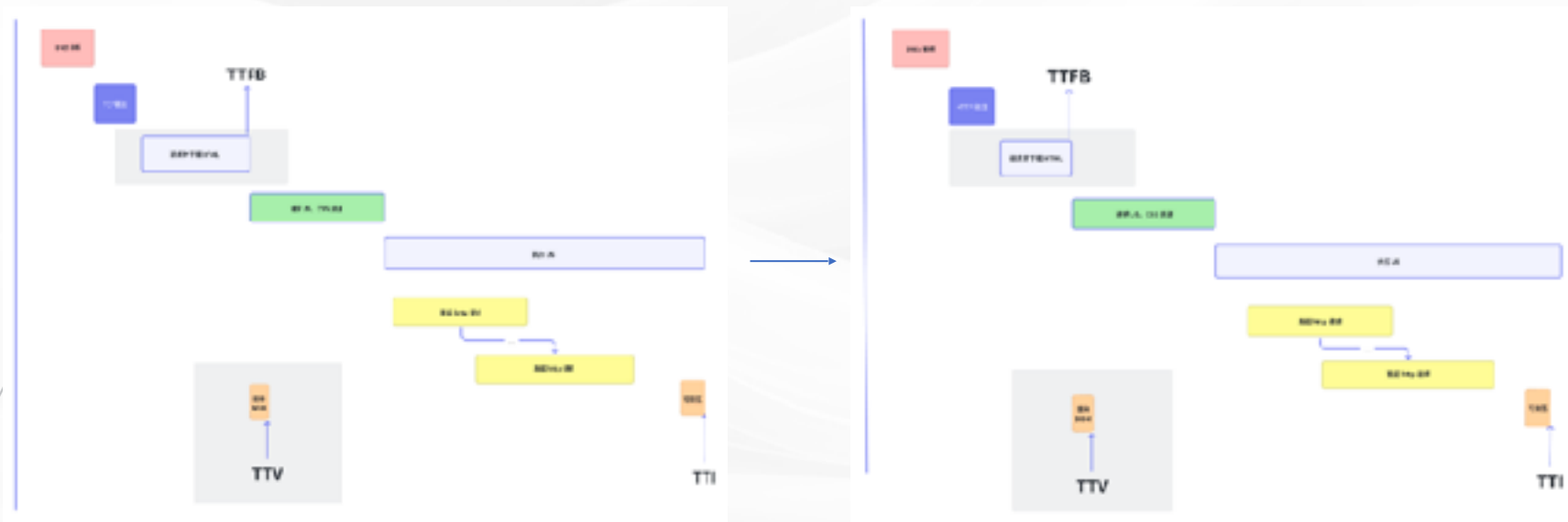
缓存陷阱 —— 热 Key

解决思路：

- 前置处理（防范于未然）：
 - 多级缓存：LRU
- 后置处理：
 - 热 key 封禁
 - 熔断

服务端侧提升——SSR 缓存优化

优化效果



服务端侧提升——流式渲染



HTTP1.1: Transfer-Encoding: chunked



HTTP2.0: 无需指定 header 即可使用

服务端侧提升——流式渲染

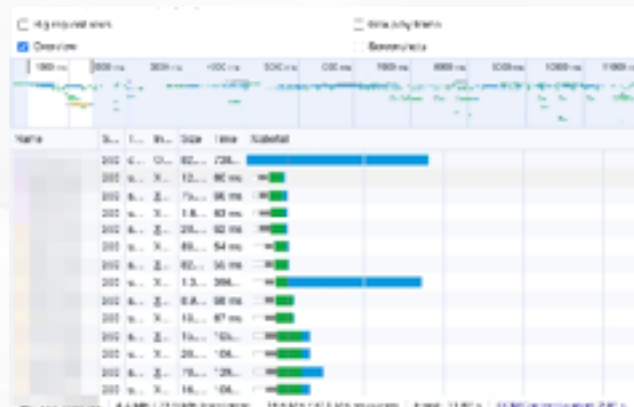
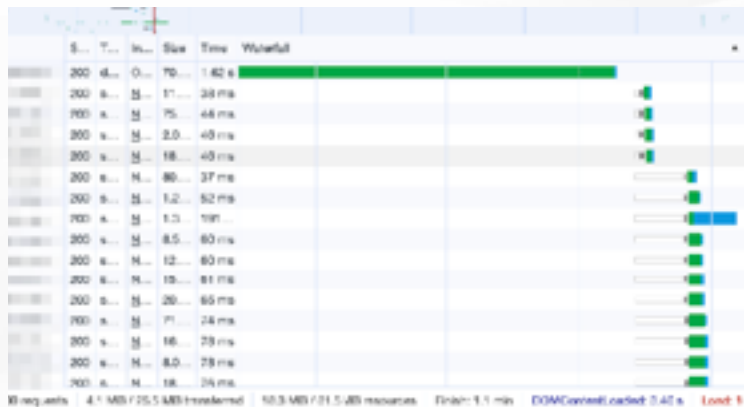


```
const html = '<html></html>'  
ctx.body = html
```



```
const htmlChunk = '<html><head></head>'  
ctx.res.write(htmlChunk)
```

服务端侧提升——流式渲染



服务端侧提升——流式渲染

优化效果

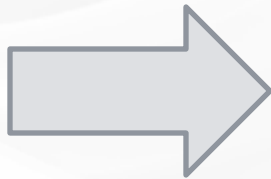


服务端侧提升——总结

- BFF、数据注入

- SSR

- 流式渲染



- TTV + TTI

- TTV

- TTFB、TTI

洞察： 页面渲染服务的定位

利用服务端执行数据获取、渲染页面的能力来达到性能优化、体验提升的目的，切勿将页面渲染服务和页面打开强耦合

洞察： 页面渲染服务的定位

WHY?

1. 服务端容器抖动
2. 服务端网络抖动
3. 强依赖下游抖动
4. 服务负载过高

服务端侧提升详解——降级

服务侧降级

数据注入降级

服务内降级

服务上层降级

数据注入降级

所有的服务端数据注入都必须有端上侧 fallback 逻辑

服务内降级

应对服务内逻辑异常，需要有可降级的模板

服务上层降级



结尾 & QA

没有银弹，适合自己的才是最好的

The image features a dark navy blue background with several thin, light blue diagonal lines. Two thicker, solid blue bars are positioned diagonally, one in the upper right and one in the lower center. The text '感谢您的观赏' is centered in a white, sans-serif font.

感谢您的观赏

早

兔年跳个好团队

早早聊 跳槽营



职业路线辅导
跳槽套路指导
模拟面试摸底
题库讲解直播
面经直播分享
优质团队内推
优质简历分享
职业成长视频

八大服务 破解 七大跳槽难题

学历硬伤
跳槽高频
项目简单
底子薄弱
深度不足
业务不精
管理不通

一起走满5年

早早聊天使票预售

三周年限定版天使年票
最后 200 张

这次,你将 TOGETHER

解锁 2020 - 2024 年大会年票会员

2020-2024 年 800 场大会直播永久观看权

晋级职场高阶能力拓展营, 40 位顶尖高手带你玩

额外的 50 场群内直播分享, 剖析业务设计思路, 现象级产品解读, 复盘经典运营活动, 分享成体系管理实践, 全方位开商业天眼, 加速技能破壁

获得 Scott 职业建议与指导 + 管家式服务

Scott 职业建议与指导 2 次, 早早聊对你的商业作品评估和投资, 跳槽服务折扣及专享招聘服务, 优质人脉共享, 助力技术成果变现...

拥有早早聊 6 年利润分红

成为早早聊天使会员, 享有 2024 ~ 2029 年盈利分红权

扫码咨询
了解更多详情

早早聊
ZAOZAO.NEED





跟我一起学习吧！

每月挑一个周六，一起充电成长！



快扫码上车早早聊

500个录播等你看

2023

全年行程

2/4	2/5	3/18	4/1	4/8	4/15	5/7	5/20	6/3	6/10	6/17	7/15	8/12	9/2	9/23	10/14	10/15	10/21	10/22	10/28	11/4	11/11	11/25	12/9	12/16	12/23
健康指南	前端规划	前端操作	杭州 GPT 沙龙	Model.js	低代码无代码	前端项目管理	上海 GPT 沙龙	求职就业新趋势	武汉 GPT 沙龙	前端性能优化	AI Chatbot	Night Of AI	前端架构	前端搞AI	技术人AI编程	表格与表单	前端跨端方案	前端工程化	玩转 Prompt	3D 互动	前端搞微前端	前端搞 Rust	前端搞浏览器	前端搞量保障	CI/CD

2022

全年行程

2021
全年
行程

页血搭建专场
小程序组件化
前端团队管理
前端自由职业

2020
全年
行程

Node.js	前端搞 IDE	前端搞 Vue	前端搞安全专场	前端搞直播专场	GraphQL 专题	前端玩转 BFF	前端搞可视化	前端搞 WebGL	前端搞彩虹桥专场	前端搞 Flutter	前端搞交互专场	前端搞职场晋升	前端搞转算法	前端搞 C/C++	大厂招聘面试	搭建新玩法	页面包建设专场	小程序组件化	前端团队管理	的前端自由职业
---------	---------	---------	---------	---------	------------	----------	--------	-----------	----------	-------------	---------	---------	--------	-----------	--------	-------	---------	--------	--------	---------