
前端性能优化
加载性能优化
懒加载（Lazy Load）：延迟加载非首屏资源，如图片、组件等。
预加载（Preload / Prefetch）：提前加载关键资源或后续页面资源
资源合并（Concatenate）：合并 CSS、JS 文件以减少请求次数
压缩资源（Gzip / Brotli）：对文本资源进行压缩，减小传输体积
使用 CDN 加速静态资源加载
启用 HTTP/2 / HTTP/3 协议：提升多资源并发加载效率
合理使用浏览器缓存（Cache-Control、ETag）
异步加载脚本：通过给标签添加async或defer属性，或者使用动态脚本注入的方式，避免JavaScript文件阻塞页面渲染。
渲染性能优化
骨架屏（Skeleton Screen）：提升用户等待时的视觉体验
服务端渲染（SSR）/ 静态生成（SSG）：加快首屏展示速度
虚拟滚动（Virtual Scrolling）：仅渲染可视区域内容，适用于长列表
减少重绘与回流（Repaint & Reflow）：批量更新样式、避免频繁操作DOM
使用防抖与节流（Debounce / Throttle）：控制高频事件触发频率
CSS 动画优化：使用 transform 和 opacity 实现 GPU 加速动画
代码与构建优化
Tree Shaking：移除未使用的 JavaScript / CSS 代码
代码拆分（Code Splitting）：按需加载模块或路由组件
按需引入第三方库（如 Lodash、Ant Design）
避免内存泄漏：及时清理定时器、事件监听器和闭包引用
封装通用组件 / 工具函数：提高复用性，减少冗余代码
图像与多媒体优化
选择合适格式（WebP / AVIF / SVG）
图片尺寸适配（响应式图片 srcset）
图片懒加载 + 占位图
使用 Web Workers 处理复杂计算任务