

1 执行摘要

本文档概述了 FlowWrite 前端设计系统的全面重构计划，旨在提升可维护性、一致性和用户体验。

2 当前问题分析

2.1 设计系统问题

2.1.1 不完整的 CSS 变量系统

1. CSS 变量 (--flow-bg、--flow-text、--flow-border 等) 被使用但未全局定义
2. 变量仅在 FlowEditor.svelte 的 .darkMode 类下定义
3. 使用不一致导致依赖内联样式
4. 缺少完整的亮色/暗色主题定义

2.1.2 主题系统缺陷

1. 深色模式切换存在但不工作 (FlowEditor.svelte:61)
2. 没有完整的明暗主题定义
3. 缺少主题 provider/store 用于状态管理

2.1.3 颜色系统问题

1. 硬编码颜色散布各处：
 - 背景色：#0f0f1a、#1a1a2e、#2d2d44
 - 文本色：#e0e0e0、#a0a0b0
 - 强调色：#6366f1、#8b5cf6、#0ea5e9
2. 缺少集中式调色板
3. 语义颜色使用不一致

2.1.4 Emoji 图标过度使用

1. 使用 20+ 个 emoji 图标 (📐、⬇️、➡️、▶️、✅、🔍、🎯、☀️、🌙、🤖 等)
2. 跨平台渲染不一致
3. 专业性不足
4. 应替换为 SVG 图标 (Lucide、Heroicons 或 Phosphor)

2.1.5 Tailwind 利用不足

1. tailwind.config.js 仅定义了 primary 颜色
2. 组件使用自定义 CSS 而非 Tailwind 工具类
3. 方法不一致：部分使用 Tailwind，部分使用自定义 CSS

2.2 代码组织问题

2.2.1 样式重复

1. 节点组件 (InputNode、OutputNode、LLMNode、TextNode) 有 70% 的 CSS 重叠
2. 按钮样式在各组件中重复
3. 缺少共享的组件基础元素

2.2.2 组件架构

1. FlowEditor.svelte (272 行) 过大，混合了 UI 和逻辑
2. ApiTest.svelte (853 行) 巨大，混合了多个关注点

3. 缺少关注点分离

2.2.3 状态管理

1. 状态分散在各组件中
2. 没有用于 UI 偏好的全局 store
3. 使用事件进行组件通信，而非 store/context

2.3 用户体验缺失

2.3.1 缺少的交互

1. 深色模式切换不起作用
2. 操作缺少加载状态
3. 使用 alert 弹窗而非合适的模态框/通知
4. 缺少键盘快捷键

2.3.2 无障碍性

1. 交互元素缺少焦点状态
2. FloatingBall 缺少适当的 ARIA 属性
3. 没有屏幕阅读器支持

2.3.3 性能

1. 内联样式在每次渲染时计算
2. 缺少组件懒加载
3. 没有 CSS 清除/优化

2.4 死代码

1. selectedNodes 和 selectedEdges 状态在 FlowEditor.svelte:59-60 未使用
2. onnodedragstop 函数在 FlowCanvas.svelte:42 为空

—

3 重构计划

3.1 阶段 1：设计系统基础（优先级：高）

3.1.1 建立 CSS 变量系统

位置：src/lib/styles/theme.css（新建）

创建全局主题 CSS 文件，定义：

1. 亮色主题变量（背景、文本、边框、语义颜色）
2. 暗色主题变量
3. 间距系统
4. 圆角系统
5. 阴影系统

操作步骤：

1. 创建全局主题 CSS 文件
2. 在 App.svelte 中导入
3. 从组件中移除硬编码颜色

3.1.2 图标系统迁移

位置： `src/lib/components/icons/`（新建目录）

安装 `lucide-svelte` 获取一致的 SVG 图标

创建图标组件：

1. `LayoutIcon.svelte`
2. `ArrowDownIcon.svelte`
3. `ArrowRightIcon.svelte`
4. `PlayIcon.svelte`
5. `CheckIcon.svelte`
6. `ZoomInIcon.svelte`
7. `ZoomOutIcon.svelte`
8. `TargetIcon.svelte`
9. `SunIcon.svelte`
10. `MoonIcon.svelte`
11. `RobotIcon.svelte`
12. `InputIcon.svelte`
13. `TextIcon.svelte`
14. `OutputIcon.svelte`

将所有 emoji 替换为图标组件，更新 `Toolbar.svelte`、`NavBar.svelte`、`NodeSidebar.svelte`

3.1.3 Tailwind 配置增强

位置： `flow-write/tailwind.config.js`

操作步骤：

1. 将 Tailwind 配置与 CSS 变量同步
2. 为 FlowWrite 添加自定义工具类
3. 启用 `class` 和 `data-attribute` 深色模式支持

3.2 阶段 2：主题系统实现（优先级：高）

3.2.1 创建主题 Provider

位置： `src/lib/stores/theme.ts`（新建文件）

使用 Svelte stores 创建主题 store：

1. `themeStore` - 管理当前主题状态
2. `setTheme()` - 设置并应用主题
3. `init()` - 从 `localStorage` 加载保存的主题
4. 支持系统偏好检测

操作步骤：

1. 创建主题 store 和持久化逻辑
2. 从 `FlowEditor.svelte` 移除 `darkMode` 状态
3. 实现明暗主题切换

3.2.2 主题切换组件

位置： `src/lib/components/ThemeToggle.svelte`（新建文件）

操作步骤：

1. 创建可重用的主题切换组件
2. 添加到 `Toolbar.svelte`
3. 确保正确的 `aria` 标签

3.3 阶段 3：组件重构（优先级：中）

3.3.1 提取共享组件

位置： `src/lib/components/primitives/`（新建目录）

创建可重用的基础组件：

1. `Button.svelte` - 标准化按钮，支持多种变体（`primary`、`secondary`、`danger`、`icon`）
2. `Card.svelte` - 具有一致样式的容器
3. `Badge.svelte` - 状态指示器
4. `Input.svelte` - 带验证的表单输入
5. `Select.svelte` - 下拉选择组件

操作步骤：

1. 创建组件基础元素
2. 在代码库中替换按钮实现
3. 确保一致的 `hover/focus` 状态

3.3.2 统一节点组件

位置： `src/lib/nodes/BaseNode.svelte`（新建文件）

创建基础节点组件，包含：

1. 共享的节点样式
2. `Handle` 位置管理
3. 图标插槽
4. 状态显示

操作步骤：

1. 创建基础节点组件
2. 扩展 `LLMNode.svelte`、`InputNode.svelte` 等以使用基础组件
3. 将 CSS 重复减少 70%
4. 确保一致的节点大小和间距

3.3.3 拆分大型组件

重构 `FlowEditor.svelte`：

1. 提取到 `FlowEditorContainer.svelte`（状态管理）
2. 提取到 `FlowEditorLayout.svelte`（布局结构）
3. 提取到 `FlowEditorToolbar.svelte`（工具栏集成）

重构 `ApiTest.svelte`：

1. 提取到 `ApiTestContainer.svelte`（逻辑）
2. 提取到 `ApiTestSettings.svelte`（设置面板）
3. 提取到 `ApiTestChat.svelte`（聊天界面）
4. 提取到 `ApiTestMessageList.svelte`（消息列表）

操作步骤：

1. 限制组件在 200 行以内
2. 分离关注点（逻辑 vs 表现）
3. 提升可测试性

3.4 阶段 4：状态管理（优先级：中）

3.4.1 创建 UI Stores

位置： `src/lib/stores/`

需要创建的文件：

1. `uiStore.ts` - 全局 UI 偏好（侧边栏宽度、面板状态）
2. `workflowStore.ts` - 超出组件的工作流状态
3. `preferencesStore.ts` - 用户偏好

操作步骤：

1. 集中管理 UI 状态
2. 启用跨组件的响应式更新
3. 移除 prop drilling

3.4.2 替换事件派发

当前： 组件派发事件，父组件监听

目标： 使用 store 或 context 进行状态共享

操作步骤：

1. 审计所有 `createEventDispatcher` 的使用
2. 用 stores 替换状态共享
3. 保留事件用于用户操作（点击、拖拽）

3.5 阶段 5：用户体验增强（优先级：中）

3.5.1 加载状态

位置： `src/lib/components>LoadingSpinner.svelte`（新建文件）

操作步骤：

1. 为异步操作添加加载旋转器
2. 为内容添加骨架加载器
3. 为工作流执行、API 调用实现

3.5.2 Toast 通知

位置： `src/lib/components/Toast.svelte` + `src/lib/stores/toast.ts`

操作步骤：

1. 用 toast 通知替换所有 `alert()` 调用
2. 添加成功/错误/信息变体
3. 超时后自动关闭

3.5.3 模态框系统

位置： `src/lib/components/Modal.svelte`（新建文件）

操作步骤：

1. 创建可重用的模态框组件
2. 添加键盘 (Escape) 支持
3. 添加背景点击关闭
4. 用于确认、设置等场景

3.5.4 键盘快捷键

位置： src/lib/utils/shortcuts.ts (新建文件)

操作步骤：

1. 添加 Ctrl+Z 撤销
2. 添加 Ctrl+Y 重做
3. 添加 Ctrl+S 保存
4. 添加 Ctrl+/ 帮助
5. 在 UI 中显示快捷键

3.6 阶段 6：无障碍性（优先级：中）

3.6.1 焦点管理

操作步骤：

1. 添加可见焦点指示器 (outline)
2. 支持键盘导航
3. 为模态框实现焦点捕获
4. 在需要处添加 tabindex

3.6.2 ARIA 属性

操作步骤：

1. 更新 FloatingBall.svelte 添加正确的 ARIA
2. 为所有按钮添加 role 和 aria-label
3. 为动态内容添加实时区域
4. 确保屏幕阅读器兼容性

3.6.3 颜色对比度

操作步骤：

1. 测试所有颜色组合是否符合 WCAG AA 标准
2. 确保文本对比度 4.5:1
3. 修复暗色模式下的对比度问题

3.7 阶段 7：性能优化（优先级：低）

3.7.1 CSS 优化

操作步骤：

1. 在 Vite 配置中启用 CSS 清除
2. 移除未使用的 Tailwind 类
3. 生产环境 CSS 压缩

3.7.2 组件优化

操作步骤：

1. 在适当位置使用 `$effect` 而非 `$derived`
2. 为子组件添加 `export let { eager } = $props()` 以实现懒加载
3. 为大列表实现虚拟滚动

3.7.3 资源优化

操作步骤：

1. SVG 图标 sprite 或 tree-shaking
2. 懒加载重型组件
3. 路由代码分割

3.8 阶段 8：清理（优先级：低）

3.8.1 移除死代码

操作步骤：

1. 从 `FlowEditor.svelte:59-60` 移除未使用的状态
2. 从 `FlowCanvas.svelte:42` 移除空的 `onnode dragstop`
3. 移除未使用的导入
4. 清理注释代码

3.8.2 代码文档

操作步骤：

1. 为复杂函数添加 JSDoc 注释
2. 文档化组件 props
3. 为非显而易见的逻辑添加内联注释

—

4 迁移时间表

阶段	持续时间	依赖关系
阶段 1：设计系统基础	2-3 天	无
阶段 2：主题系统	1-2 天	阶段 1
阶段 3：组件重构	3-4 天	阶段 1、2
阶段 4：状态管理	2-3 天	阶段 1、2
阶段 5：UX 增强	2-3 天	阶段 3、4
阶段 6：无障碍性	1-2 天	阶段 3、5
阶段 7：性能优化	1-2 天	所有阶段
阶段 8：清理	1 天	所有阶段

预计总时间：13-20 天

—

5 成功指标

1. 所有硬编码颜色替换为 CSS 变量
2. 深色模式完全可用
3. Emoji 图标替换为 SVG 图标
4. 代码重复减少 50%
5. 所有组件小于 200 行
6. 所有 `alert()` 调用替换为 `toasts/模态框`
7. 符合 WCAG AA 无障碍标准
8. 页面加载时间 < 1 秒
9. Lighthouse 评分 > 90

—

6 待解决问题

1. 是否应该使用 UI 组件库（shadcn-svelte、Skeleton UI）还是自定义？
 2. 图标库选择：Lucide、Heroicons 还是 Phosphor？
 3. 动画库选择：Framer Motion 还是 Svelte Motion？
 4. 是否需要添加 i18n 支持以实现国际化？
-

7 资源

1. Svelte 5 Runes 文档：<https://svelte.dev/docs/runes>
 2. Tailwind CSS 主题：<https://tailwindcss.com/docs/dark-mode>
 3. Web 内容无障碍指南 (WCAG)：<https://www.w3.org/WAI/WCAG21/quickref/>
 4. Lucide Icons：<https://lucide.dev/>
-

最后更新：2025-01-21