组件说明

核心组件

App组件

```
// App.tsx
// 应用程序的根组件，负责整体布局和路由
interface AppProps {}

const App: React.FC<AppProps> = () => {
  // 布局结构：Header + (ElementPanel + Canvas + PropertyPanel)
};
```

Canvas组件

```
// components/Canvas/index.tsx
// 画布容器组件，基于Ant Design布局系统
interface CanvasProps {}

const Canvas: React.FC<CanvasProps> = () => {
  // 集成Redux状态管理
  // 处理视图模式切换逻辑
};

// components/Canvas/Canvas2D.tsx
// 基于Canvas 2D的平面绘图引擎
interface Canvas2DProps {}

const Canvas2D: React.FC<Canvas2DProps> = () => {
  // 实现网格绘制、图元渲染
  // 处理选择、移动等交互逻辑
  // 集成几何计算工具
};

// components/Canvas/Canvas3D.tsx
// Three.js驱动的3D渲染组件
interface Canvas3DProps {}

const Canvas3D: React.FC<Canvas3DProps> = () => {
  // Three.js场景初始化与管理
  // 3D图元实例化渲染
  // 相机控制与灯光设置
};
```

面板组件

```
// components/ElementPanel/index.tsx
// 元素选择面板
interface ElementPanelProps {}

const ElementPanel: React.FC<ElementPanelProps> = () => {
  // 提供图元库和创建工具
};

// components/PropertyPanel/index.tsx
// 属性编辑面板
interface PropertyPanelProps {
  selectedElement: BaseElement | null;
  onUpdate: (element: BaseElement) => void;
  onDelete: (elementId: string) => void;
  scale: number;
  coordinateSystem: '2d' | '3d';
}

const PropertyPanel: React.FC<PropertyPanelProps> = () => {
  // 编辑选中图元的属性
};
```

图元组件

基础图元

```
// components/Elements/BaseElement.tsx
// 图元基类组件
interface BaseElementProps {
  element: BaseElement;
  selected: boolean;
  hovered: boolean;
  onSelect: (id: string) => void;
  onHover: (id: string | null) => void;
}
```

墙体组件

```
// components/Elements/WallElement.tsx
interface WallElementProps extends BaseElementProps {
  element: WallElement;
}

const WallElement: React.FC<WallElementProps> = () => {
```

```
  // 渲染墙体
  // 处理墙体交互
};
```

座位组件

```
// components/Elements/SeatElement.tsx
interface SeatElementProps extends BaseElementProps {
  element: SeatElement;
}

const SeatElement: React.FC<SeatElementProps> = () => {
  // 渲染座位
  // 处理座位交互
};
```

门窗组件

```
// components/Elements/DoorElement.tsx
interface DoorElementProps extends BaseElementProps {
  element: DoorElement;
}

const DoorElement: React.FC<DoorElementProps> = () => {
  // 渲染门
  // 处理门的交互
};

// components/Elements/WindowElement.tsx
interface WindowElementProps extends BaseElementProps {
  element: WindowElement;
}

const WindowElement: React.FC<WindowElementProps> = () => {
  // 渲染窗
  // 处理窗的交互
};
```

工具组件

文件对话框

```
// components/FileDialog/index.tsx
interface FileDialogProps {
  visible: boolean;
```

```
  mode: 'open' | 'save';
  onClose: () => void;
  onSelect: (file: FileListItem) => void;
}

const FileDialog: React.FC<FileDialogProps> = () => {
  // 文件列表展示
  // 文件操作处理
};
```

工具栏

```
// components/Header/index.tsx
interface HeaderProps {}

const Header: React.FC<HeaderProps> = () => {
  // 文件操作
  // 编辑操作
  // 视图切换
  // 用户操作
};
```

辅助组件

捕捉提示

```
// components/Canvas/SnapGuide.tsx
interface SnapGuideProps {
  snapPoints: SnapPoint[];
  alignmentGuides: AlignmentGuide[];
}

const SnapGuide: React.FC<SnapGuideProps> = () => {
  // 渲染捕捉点
  // 渲染对齐参考线
};
```

尺寸标注

```
// components/Canvas/Dimension.tsx
interface DimensionProps {
  startPoint: Point;
  endPoint: Point;
  offset: number;
}
```

```tsx
const Dimension: React.FC<DimensionProps> = () => {
  // 渲染尺寸标注
};
```

网格背景

```tsx
// components/Canvas/Grid.tsx
interface GridProps {
  size: number;
  enabled: boolean;
}

const Grid: React.FC<GridProps> = () => {
  // 渲染网格
};
```

组件通信

状态管理

- 使用Redux管理全局状态
- 使用React Context管理局部状态
- 使用Props传递组件间数据

事件处理

- 使用自定义事件系统处理Canvas交互
- 使用Redux Action处理状态更新
- 使用回调函数处理组件间通信

性能优化

- 使用React.memo优化组件重渲染
- 使用useMemo缓存计算结果
- 使用useCallback缓存回调函数