# Element-plus

## 1.1更换主题

①利用sass变量覆盖组件默认样式

②利用css变量覆盖组件默认样式

优点：可以精确覆盖所有组件的样式且不会产生样式冲突，针对更换固定主题时，可通过工具生成所有样式文件

缺点：自定义程度较高的更换主题时（例如明亮主题嵌套暗黑主题、组件带有渐变色），需要抽离所有组件样式，且一套样式只能适用一种场景，公共样式会影响所有组件



## 1.2暗黑模式

使用更换主题的方式，重写一套dark模式的样式，通过在body上挂载dark类名实现暗黑模式

## 1.3总结

若项目定制化程度较小，切换主题只设置为固定主题可采用。出现特殊定制化时，需针对定制化的地方自定义样式且一套样式只适用一种场景，其次，每个组件拥有的样式不仅局限于自身，可能同时引用了公共的全局样式，修改某个组件样式可能对其他组件产生影响，组件本身涉及到的样式多，抽离组件本身的样式成本较大，定制化成本很高

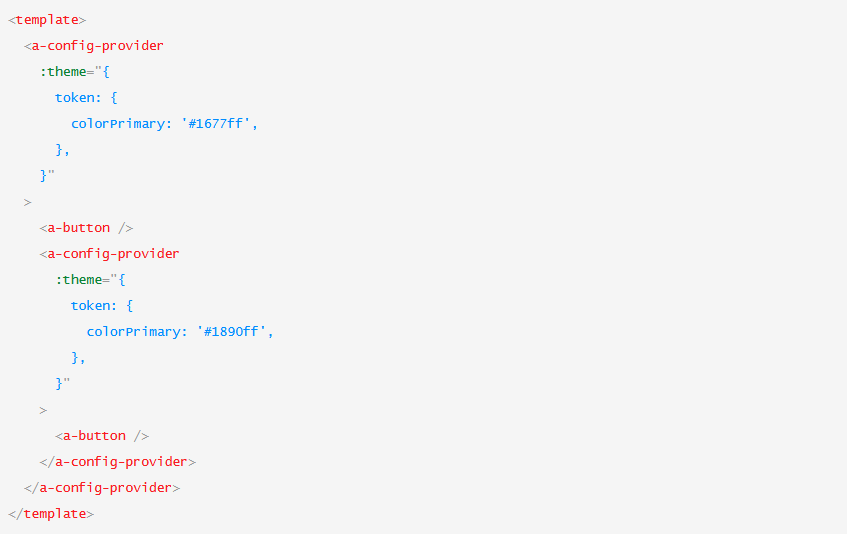
# Ant Design Vue

## 2.1更换主题

①采用css in js方案更换主题

优点：精确覆盖组件样式，且组件与组件之间有样式隔离，可控制公共样式不会影响到所有组件。公共样式拥有样式token，内置算法基于seedToken生成主题，使用起来灵活

缺点：seedToken基础配置繁多（定制化需要较高时），组件之间若要实现样式隔离需要在开发时手动配置，若定制化的组件多，维护样式隔离的成本较大



2.2暗黑模式

更改内置算法为暗黑模式即可（支持三种模式：基础（亮色）、暗色、紧凑）

2.3总结

切换主题时拥有组件样式隔离，能避免影响全局样式，内置算法可以很方便的切换模式。但是使用组件之间样式隔离会带来开发成本且后续也有维护成本，内置的算法不可更改，有自己的主题编辑器。

# Naive UI

## 3.1更换主题

①利用NConfigProvider全局配置更换主题

优点：不同组件可以有样式隔离，并且组件的所有样式以及属性均可以配置 缺点：不配置时通过默认主题算法给组件样式，配置时若精确控制组件样式，需要覆盖的样式会比较多





3.2暗黑模式

需重写全局配置（若无自定义组件样式，可直接切换暗黑模式）

3.3 总结

针对单一模式可以很方便的配置组件样式，且可定制化程度极高，能实现组件之间样式隔离，能满足大多数场景下的主题定制功能。

# arco-design-vue

## 4.1更换主题

①通过less变量覆盖组件默认样式

优点：精确控制组件样式，最重要的是拥有组件样式风格配置平台，可以将样式打包后直接引入，开发无维护成本（由UI维护组件样式），且可以配置多套样式风格直接切换

缺点：自定义组件无法响应主题切换，需额外处理，风格配置平台无法配置渐变色。

## 4.2暗黑模式

通过风格配置平台的主题包可直接切换至暗黑模式

## 4.3总结

利用风格配置平台可以做到100%还原ui的设计稿样式（渐变色配置除外），且直接支持暗黑模式的切换，开发无任何维护成本，能满足大多数情况下的主题定制功能。

# 综合对比

主题切换技术：基于sass、less、css、css in js 均可以实现通过变量切换主题

主题切换成本：1.Element plus需要抽离组件样式，且无法做到组件样式隔离，抽离样式成本高，暗黑模式需要重写样式。2.Ant design vue基于seedToken生成组件样式，能做到样式隔离且直接支持暗黑模式，配置Token（基于seedToken的衍生token）繁琐 3. Naiva Ui能做到定制化每个组件的样式且组件样式隔离，配置configProvide繁琐，且暗黑模式需要重写配置。4.Arco Design Vue支持组件样式隔离，能通过风格配置平台配置，可直接支持暗黑模式，风格配置平台配置繁琐，但是可以支持多套主题。

综上 Arco > Naive ≈ Ant > Element