

字节T6外包笔试题 副本

T6

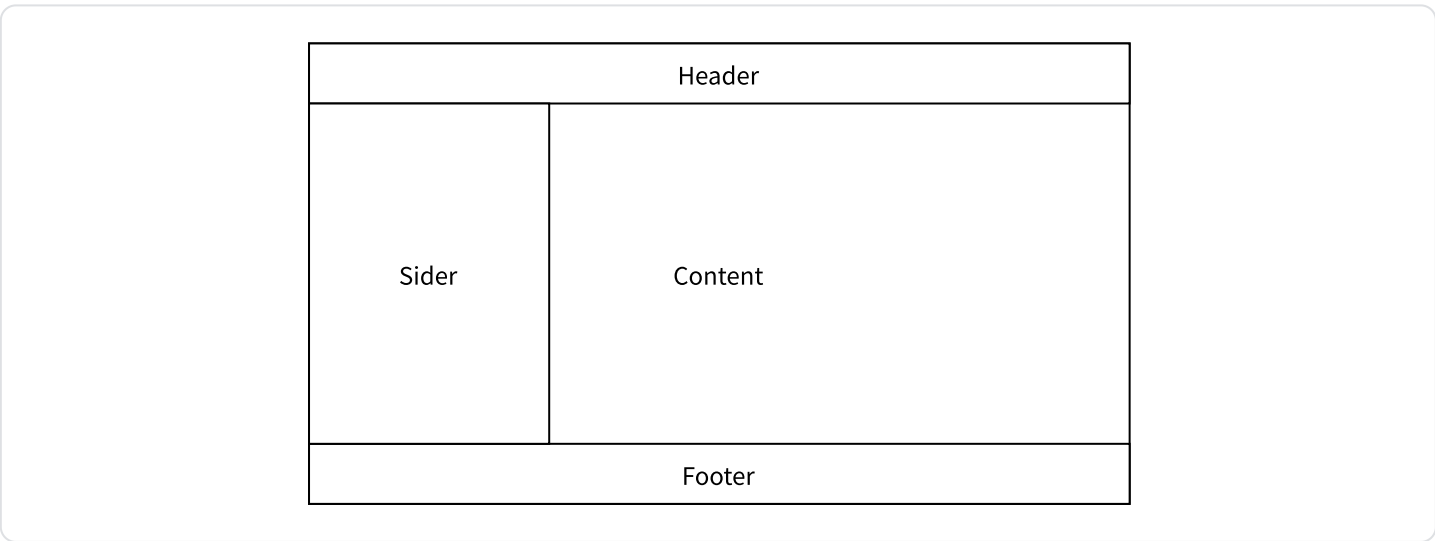
编程题目

- 设计一个函数能将字符串转换成小驼峰格式
- 假设字符串的分隔符只有" ", "-", "_"三种

```
1 camelCase('Foo Bar');
2 // => 'fooBar'
3
4 camelCase('foo-bar');
5 // => 'fooBar'
6
7 camelCase('FOO_BAR');
8 // => 'fooBar'
9
10 function camelCase(str) {
11     if (!str && !str.trim()) return "请输入正确的字符串"
12     let operator = " "
13     if (/\/-+/g.test(str)) operator = "-"
14     if (/\/_+/g.test(str)) operator = "_"
15     const [a, ...b] = str.split(operator)
16     const firstStr= a.toLocaleLowerCase()
17     const otherStr= b.map(v =>`${v.slice(0, 1).toLocaleUpperCase()}${v.slice(1
18         ).toLocaleLowerCase()}`)
19     return [firstStr, ...otherStr].reduce((pre, cur) => `${pre}${cur}`, "")
20 }
```

组件设计

- 实现一套Layout容器组件, 包括Header,Footer,Sider,Content四种子组件



- 要求:
 - Header与Footer分别在Layout容器顶部和底部, 宽度占满全屏, 高度均为50px
 - Sider是否展示通过props可以配置, 如果展示的话宽度为50px, 高度占满剩余空间

- Content占满剩下所有空间
- 通过React或者Vue实现: 用户可以在Header/Content/Footer中自由填入内容

```

1 // 实现组件功能及样式
2 const Header = () => {}
3 const Footer = () => {}
4 const Content = () => {}
5 const Layout = () => {}
6
7 // 使用场景
8 const App = () => (
9   <Layout hasSider={true}>
10     <Header>我是头</Header>
11     <Content>我是身子</Content>
12     <Footer>我是尾</Footer>
13   </Layout>
14 )

```

```

1 /**-----Layout-----
2 */
3 import {
4   useMemo,
5   useCallbak,
6   MapHTMLAttributes,
7   ReactElement,
8   CSSProperties
9 } from 'react'
10 export interface LayoutProps extends MapHTMLAttributes<ReactElement> {
11   hassider?: boolean
12 }
13 type LayoutChildrenName = 'Header' | 'Content' | 'Footer'
14 export const Layout = (props: LayoutProps): JSX.Element => {
15   const { hassider } = props
16   const layOutStyle: CSSProperties = {
17     height: '100vh',
18     display: 'flex',
19     flexDirection: 'column'
20   }
21   const siderStyle: CSSProperties = {
22     flexBasis: '50px',
23     flexGrow: 0,
24     backgroundColor: 'green'
25   }
26
27   const renderDom = useCallbak(
28     (type: LayoutChildrenName) => {
29       if (!props.children) return 'Empty'
30       if (Array.isArray(props.children)) return props.children.find(v =>
31         v.type.name === type) || null
32       if (typeof props.children === 'object' && (props.children as any).t
33         ype.name === type) return props.children
34       return null
35     },
36     [props.children]
37   )

```

```

36     const renderSider = useMemo(() => hassider ? (
37         <div className="sider" style={siderStyle}> sider</div>
38     ) : null,[hassider, siderStyle])
39     return (
40         <div style={layOutStyle}>
41             {renderDom('Header')}
42             <div style={{ display: 'flex', flexGrow: 1 }}>
43                 {renderSider}
44                 {renderDom('Content')}
45             </div>
46             {renderDom('Footer')}
47         </div>
48     )
49 }
50
51 /**-----Header-----
52 */
53
54 import { MapHTMLAttributes, ReactElement,CSSProperties } from 'react'
55
56 export interface HeaderPorps extends MapHTMLAttributes<ReactElement> {}
57 export const Header = (props: HeaderPorps): JSX.Element => {
58     const style: CSSProperties = {
59         display: 'flex',
60         height: '50px',
61         backgroundColor: 'skyblue'
62     }
63     return <div style={style}>{props.children}</div>
64 }
65
66 /**-----Content-----
67 */
68
69 import { MapHTMLAttributes, ReactElement,CSSProperties } from 'react'
70
71 export interface ContentPorps extends MapHTMLAttributes<ReactElement> {}
72 export const Content = (props: ContentPorps): JSX.Element => {
73     const style: CSSProperties = {
74         display: 'flex',
75         flexGrow: 1,
76         backgroundColor: '#ccc',
77     }
78     return <div style={style}>{props.children}</div>
79 }
80
81 /**-----Footer-----
82 */
83
84 import { MapHTMLAttributes, ReactElement, CSSProperties } from 'react'
85
86 export interface FooterPorps extends MapHTMLAttributes<ReactElement> {}
87 export const Footer = (props: FooterPorps): JSX.Element => {
88     const style: CSSProperties = {
89         display: 'flex',
90         height: '50px',
91         backgroundColor: 'skyblue',
92     }
93     return <div style={style}>{props.children}</div>
94 }
95

```

