



React + typescript + antd开发前端应用（七）添加菜单点击功能



ep76

关注

上一篇：[React + typescript + antd开发前端应用（六）应用基本框架组件](#)

完成基本框架的开发后，接下来就要逐步完善点击菜单后在Tabs组件中添加新的Tab页的功能。仔细分析页面源码，发现AppTabs组件的标签页数据是一个state中的数组，只要在点击菜单是，能够项这个数组中添加Tab页所需的数据就可以实现添加Tab的功能。而点击事件则是发生在AppMenu组件中。就是说，两个兄弟组件之间（AppTabs和AppMenu）需要相互传递数据。可以根据[React + typescript + antd开发前端应用（四）代码拆分](#)的知识，需要把数据定义迁移到AppLayout组件中，然后通过两个组件中传递state、函数等完成数据修，从而达到修改应用Dom结构的目的。通过干部state中的数据而达到修改页面外观，就是现代前端框架所谓的响应式设计，是新一代前端框架追求的目的。这一点对可能习惯于使用jQuery这种直接操作Dom的开发人员来说，可能还不是很好理解，只能通过多写代码来熟悉这种通过修改数据进而完成Dom修改的新一代前端框架。

1、创建AppLayout组件的行为组件

创建AppTabs组件对应的行为组件src\layout\AppLayoutFuncs.ts，编辑文件内容如下：

```
//定义Tabs的每个标签页对象的数据结构
export type TabItem = { label: string, children: string, key: string, closable?: boole
//定义Tabs的数组，并提供首页页签的默认数据
export const initialTabItems = [
  { label: '首页', children: '首页页签内容', key: '1', closable: false }
];
/**
 * 选中某菜单项时调用函数，在Tabs中添加新的Tab
 * @param menuKeyId 选中的菜单数据keyId
 * @param items state中的Tab对象数组
 * @param setActiveKey 修改Tabs选中状态的Tabid的函数
 * @param setItems 修改state中的Tab对象数组的函数
 */
export function selectedMenuKey(menuKeyId: string, items: TabItem[], setActiveKey: (ne
  addNewTab(menuKeyId, '功能: ' + menuKeyId, items, setActiveKey, setItems);
}
/**
 * 添加新的Tab: 如果点击菜单对应的Tab页签已经存在，则将Tab页签设置为选中状态，否则添加新的页签并
 * @param tabKey 新Tab对象的key
 * @param tabLabel 新Tab对象的Title文字
 * @param items state中的Tabs数组
 * @param setActiveKey 修改Tabs选中状态的Tabid的函数
 * @param setItems 修改state中的Tab对象数组的函数
 */
export function addNewTab(tabKey: string, tabLabel: string, items: TabItem[], setActiv
  if (items.some((oneItem) => oneItem.key === tabKey)) {
    setActiveKey(tabKey);
  } else {
    const newPanes = [...items];
    newPanes.push({ label: tabLabel, children: tabKey + ", Tab内容就是: " + tabLabel:
    setItems(newPanes);
    setActiveKey(tabKey);
```

赞同



添加评论

分享

喜欢

收藏

申请转载





```
    }
  };
};
```

2、创教AppTabs组件的行为组件

创建src\layout\AppTabsFuncs.ts文件，编辑文件内容如下：

```
import { TabItem } from './AppLayoutFuncs';

//标签页点击事件对象
export type TargetKey = React.MouseEvent | React.KeyboardEvent | string;
//<AppTabs>标签需要接收父组件的的数据对象
export type Param = {
  setActiveKey: (newKeyId: string) => void,
  items: TabItem[],
  setItems: (itemArray: TabItem[]) => void,
  activeKey: string
};
/**
 * 点击某页签标题时：将点击的Tab页签状态改为设置状态
 * @param newActiveKey 点击页签的key
 * @param props 父组件传递过来的数据对象
 */
export function onChange (newActiveKey: string, props: Param) {
  props.setActiveKey(newActiveKey);
};
/**
 * 点击某页签的关闭按钮时触发：
 * @param targetKey 被关闭页签的id
 * @param props 父组件传递过来的数据对象
 */
export function remove(targetKey: TargetKey, props: Param) {
  let newActiveKey = props.activeKey;
  let lastIndex = -1;
  props.items.forEach((item, i) => {
    if (item.key === targetKey) {
      lastIndex = i - 1;
    }
  });
  const newPanes = props.items.filter((item) => item.key !== targetKey);
  if (newPanes.length && newActiveKey === targetKey) {
    if (lastIndex >= 0) {
      newActiveKey = newPanes[lastIndex].key;
    } else {
      newActiveKey = newPanes[0].key;
    }
  }
  props.setItems(newPanes);
  props.setActiveKey(newActiveKey);
};
/**
 * 点击某页签的关闭按钮时触发
 * @param targetKey 被关闭的Tab的key
 * @param action 具体动作（add或者remove，由于因此了Tabs的添加按钮，因此onEdit事件只有remove
 * @param props 父组件传递过来的数据对象
 */
export function onEdit(targetKey: React.MouseEvent | React.KeyboardEvent | string, act
  if (action === 'remove') {
    remove(targetKey, props);
```

```
    }
  };
};
```

3、传教AppMenu组件的行为组件

创建src\layout\AppMenuFuncs.ts文件，修改内容如下：

```
import type { MenuProps } from 'antd';
import { TabItem } from './AppLayoutFuncs';

//定义每个菜单项的数据对象
export type MenuItem = Required['items'][number];
/**
 * 根据参数生成菜单项对象
 * @param label 菜单项目显示文字
 * @param key 菜单项唯一key
 * @param icon 菜单项图标
 * @param children 子菜单数组
 * @param type 菜单类型，由子菜单是值为group
 * @returns 返回菜单项实例对象
 */
export function getItem(label: React.ReactNode, key: React.Key, icon?: React.ReactNode
  //等同于: return {key: key, icon: icon, children: children, label: label, type: typ
  return {key, icon, children, label, type} as MenuItem;
}
//菜单初始化数据数组
export const menuItems: MenuProps['items'] = [
  getItem('系统菜单', 'systemMenu', null, [
    getItem('功能一', 'systemMenu_menu1'),
    getItem('功能二', 'systemMenu_menu2'),
    getItem('功能三', 'systemMenu_menu3')
  ])
];
//<AppMenu>组件所需父组件传递的数据类型
export type Param = {
  selectedMenuKey: (keyId: string, items: TabItem[], setActiveKey: (newKeyId: string
  setActiveKey: (newKeyId: string) => void,
  items: TabItem[],
  setItems: (itemArray: TabItem[]) => void,
};
```

4、修改AppLayout组件

修内容主要是引入行为组件和创建必要的state数据，代码如下：

```
import { useState } from 'react';
import { Layout, ConfigProvider, theme } from "antd";
import './AppLayout.css';
import AppMenu from "../AppMenu";
import AppTabs from "../AppTabs";
import { selectedMenuKey, TabItem, initialTabItems } from "../AppLayoutFuncs";
const { Header, Sider, Content } = Layout;

function AppLayout() {
  const [activeKey, setActiveKey] = useState(initialTabItems[0].key);
  const [tabItems, setTabItems] = useState<TabItem[]>(initialTabItems);
  return (
    <ConfigProvider them
```

```

    <Layout>
      <Header>Header部分</Header>
      <Layout>
        <Sider width={190} style={{overflow: "auto"}}>
          <AppMenu selectedMenuKey={selectedMenuKey} setActiveKey={setActiveKey} />
        </Sider>
        <Content>
          <AppTabs activeKey={activeKey} setActiveKey={setActiveKey} />
        </Content>
      </Layout>
    </Layout>
  </ConfigProvider>
);
};

export default AppLayout;

```

5、修改AppTabs组件

修改后的代码如下：

```

import { Tabs } from 'antd';
import { Param, onChange, onEdit } from './AppTabsFuncs';

function AppTabs(props: Param) {
  return (
    <Tabs
      type="editable-card"
      onChange={(newKeyId) => {onChange(newKeyId, props) }}
      activeKey={props.activeKey}
      onEdit={(targetKey, action) => {onEdit(targetKey, action, props)}}
      items={props.items}
      hideAdd={true}
      defaultActiveKey='1'
    />
  );
};

export default AppTabs;

```

6、修改AppMenu组件

代码如下：

```

import type { MenuProps } from 'antd';
import { Menu } from 'antd';
import { Param } from './AppMenuFuncs';
import { menuItems } from './AppMenuFuncs';

function AppMenu(props: Param) {
  const onClick: MenuProps['onClick'] = (e) => {
    props.selectedMenuKey(e.key, props.items, props.setActiveKey, props.setItems);
  };
  return (
    <Menu onSelect={onClick} defaultOpenKeys={['systemMenu']} mode="inline" items=
    />
  );
}

```

```
export default AppMenu;
```

这节内容主要是代码重构，而且需要重构的代码比较多，建议读者在重构组件过程中，不要复制教程中的源码后粘贴使用，然后进行最终结果的验证。强烈建议读者能够以教程中的代码为参考，采用自己的节奏逐步把这些函数、变量定义等内容迁移到正确的位置，而且在迁移过程中随时查看运行结果，这个过程可能会比较缓慢，但是过程中我们能体会到很多数据传递、函数传递、函数调用等等方面的细节知识，毕竟有句话叫“魔鬼都隐藏在细节中”，这句话不是没有道理啊。

编辑于 2024-02-25 06:47 · IP 属地上海


ReactTypeScriptantd

欢迎参与讨论

发布

还没有评论，发表第一个评论吧

推荐阅读



从零开始配置 react + typescript（一）：dotfiles

余腾靖

Task Tracker

Doctors Appointment

September 1st at 2:30pm

Meeting at Scholl

September 3rd at 1:30pm

Food Shopping

September 3rd at 11:00am

Add Task

用 ReScript & React 写一个 Todo App

卢米安Lumine

TypeScript React Starter

这个快速入门指南将教你如何使用 React连接TypeScript。最后，将会获得：一个使用React和TypeScript的项目用TSLint项目检查用Jest和Enzyme进行测试，Redux流程管理我们将使用 creat...

sunsh...发表于大前端工程...

入门

本入门Type下信!的项!用Jes做状

catki

https://zhuanlan.zhihu.com/p/659710221

5/5