**import** React **from 'react'  
import** { render } **from 'react-dom'  
import** { createStore } **from 'redux'  
import** { **Provider** } **from 'react-redux'  
import** *App* **from './components/App'  
import** reducer **from './reducers'  
  
const** store = createStore(reducer)  
  
render(  
 <**Provider store=**{store}>  
 <**App** />  
 </**Provider**>,  
 **document**.getElementById(**'root'**)  
)

///////////////////////////////////////////////////////////////////////////////////

**import** React **from 'react'  
import** *Footer* **from './Footer'  
import AddTodo from '../containers/AddTodo'  
import** VisibleTodoList **from '../containers/VisibleTodoList'  
  
const** App = () => (  
 <**div**>  
 <**AddTodo** />  
 <**VisibleTodoList** />  
 <**Footer** />  
 </**div**>  
)  
  
**export default** App

/////////////////////////////////////////////////////////////////////////////////////////

**import** React **from 'react'  
import** { **connect** } **from 'react-redux'  
import** { *addTodo* } **from '../actions'  
  
let** *AddTodo* = ({ dispatch }) => {  
 **let** input  
  
 **return** (  
 <**div**>  
 <**form onSubmit=**{e => {  
 e.preventDefault()  
 **if** (!input.**value**.trim()) {  
 **return** }  
 dispatch(*addTodo*(input.**value**))

//dispatch 发出一个action  
 input.**value** = **''** }}>  
 <**input ref=**{node => {  
 input = node  
 }} />  
 <**button type="submit"**>  
 Add ***Todo*** </**button**>  
 </**form**>  
 </**div**>  
 )  
}  
*AddTodo* = **connect**()(*AddTodo*)  
  
**export default** *AddTodo*

*/////////////////////////////////////////////////////////////*

*actions/index.js 里面是事件发生器函数*

**let** nextTodoId = 0  
**export const** *addTodo* = (text) => ({  
 **type**: **'ADD\_TODO'**,  
 **id**: nextTodoId++,  
 text  
})  
  
**export const** *setVisibilityFilter* = (filter) => ({  
 **type**: **'SET\_VISIBILITY\_FILTER'**,  
 filter  
})  
  
**export const** *toggleTodo* = (id) => ({  
 **type**: **'TOGGLE\_TODO'**,  
 id  
})

//////////////////////////////////////////////////////////////////

todoList 元件

**import** { **connect** } **from 'react-redux'  
import** { *toggleTodo* } **from '../actions'  
import** *TodoList* **from '../components/TodoList'  
  
const** *getVisibleTodos* = (todos, filter) => {  
 **switch** (filter) {  
 **case 'SHOW\_ALL'**:  
 **return** todos  
 **case 'SHOW\_COMPLETED'**:  
 **return** todos.filter(t => t.**completed**)  
 **case 'SHOW\_ACTIVE'**:  
 **return** todos.filter(t => !t.**completed**)  
 **default**:  
 **throw new** Error(**'Unknown filter: '** + filter)  
 }  
}  
  
**const** *mapStateToProps* = (state) => ({  
 **todos**: *getVisibleTodos*(state.**todos**, state.*visibilityFilter*)  
})  
  
**const** mapDispatchToProps = ({  
 **onTodoClick**: *toggleTodo*})  
  
**const VisibleTodoList = connect(  
 *mapStateToProps*,  
 mapDispatchToProps  
)(*TodoList*)**  
**export default** VisibleTodoList

/////////////////////////////////////////////////////////////////////////////////////////////////////

**import** React, { PropTypes } **from 'react'  
  
const** *Todo* = ({ onClick, completed, text }) => (  
 <**li  
 onClick=**{onClick}  
 **style=**{{  
 **textDecoration**: completed ? **'line-through'** : **'none'** }}  
 >  
 {text}  
 </**li**>  
)  
  
*Todo*.**propTypes** = {  
 **onClick**: PropTypes.**func**.isRequired,  
 **completed**: PropTypes.**bool**.**isRequired**,  
 **text**: PropTypes.**string**.**isRequired**}  
  
**export default** *Todo*

///////////////////////////////////////////////////////////////////////////////////////////////

**import** React, { PropTypes } **from 'react'  
import** *Todo* **from './Todo'  
  
const** *TodoList* = ({ *todos*, onTodoClick }) => (  
 <**ul**>  
 {*todos*.**map**(todo =>  
 <**Todo  
 key=**{todo.**id**}  
 {...todo}  
 **onClick=**{() => onTodoClick(todo.**id**)}  
 />  
 )}  
 </**ul**>  
)  
  
*TodoList*.**propTypes** = {  
 **todos**: PropTypes.*arrayOf*(PropTypes.*shape*({  
 **id**: PropTypes.**number**.**isRequired**,  
 **completed**: PropTypes.**bool**.**isRequired**,  
 **text**: PropTypes.**string**.**isRequired** }).**isRequired**).**isRequired**,  
 **onTodoClick**: PropTypes.**func**.isRequired  
}  
  
**export default** *TodoList*

/////////////////////////////////////////////////////////////////////////////////////////

footer.js

**import** React **from 'react'  
import** FilterLink **from '../containers/FilterLink'  
  
const** *Footer* = () => (  
 <**p**>  
 Show:  
 {**" "**}  
 <**FilterLink filter="SHOW\_ALL"**>  
 All  
 </**FilterLink**>  
 {**", "**}  
 <**FilterLink filter="SHOW\_ACTIVE"**>  
 Active  
 </**FilterLink**>  
 {**", "**}  
 <**FilterLink filter="SHOW\_COMPLETED"**>  
 Completed  
 </**FilterLink**>  
 </**p**>  
)  
  
**export default** *Footer*

////////////////////////////////////////////////////////////////////////////////////////////////

另一套

**import** React, { PropTypes } **from 'react'  
  
const** *Link* = ({ active, children, onClick }) => {  
 **if** (active) {  
 **return** <**span**>{children}</**span**>  
 }  
  
 **return** (  
 <**a href="#"  
 onClick=**{e => {  
 e.preventDefault()  
 onClick()  
 }}  
 >  
 {children}  
 </**a**>  
 )  
}  
  
*Link*.**propTypes** = {  
 **active**: PropTypes.**bool**.**isRequired**,

//由state改变从reducer 中传回来

*//mapStateToProps*  
 **children**: PropTypes.**node**.**isRequired**,

//具体显示的内容  
 **onClick**: PropTypes.**func**.isRequired

//通过点击 dispatch一个action

*// mapDispatchToProps*  
}  
  
**export default** *Link*

**import** { **connect** } **from 'react-redux'  
import** { *setVisibilityFilter* } **from '../actions'  
import** *Link* **from '../components/Link'  
  
const** *mapStateToProps* = (state, ownProps) => ({  
 **active**: ownProps.filter === state.*visibilityFilter*})  
  
**const** *mapDispatchToProps* = (dispatch, ownProps) => ({  
 onClick: () => {  
 dispatch(*setVisibilityFilter*(ownProps.filter))  
 }  
})  
  
**const** FilterLink = **connect**(  
 *mapStateToProps*,  
 *mapDispatchToProps*)(*Link*)  
  
**export default** FilterLink

dispatch的action

**export const** *setVisibilityFilter* = (filter) => ({  
 **type**: **'SET\_VISIBILITY\_FILTER'**,  
 filter  
})

//////////////////////////////////////////////////////////////////////////////

对应的reducer 函数

**const** *visibilityFilter* = (state = **'SHOW\_ALL'**, action) => {  
 **switch** (action.**type**) {  
 **case 'SET\_VISIBILITY\_FILTER'**:  
 **return** action.filter  
 **default**:  
 **return** state  
 }  
}  
  
**export default** *visibilityFilter*

reducer处理之后对应 *mapStateToProps*

**const** *mapStateToProps* = (state, ownProps) => ({  
 **active**: ownProps.filter === state.*visibilityFilter*})

返回一个active

然后最初始的元件Link 根据 active判断是否显示

**const** *Link* = ({ active, children, onClick }) => {  
 **if** (active) {  
 **return** <**span**>{children}</**span**>  
 }