



```
65  componentDidMount() {
66     document.body.addEventListener( type: 'click', this.handleBodyClick, useCapture: false)
67
68     this.input.addEventListener( type: 'click', this.handleInputClick, useCapture: false)
69 }
70
71  componentWillUnmount() {
72     document.body.removeEventListener( type: 'click', this.handleBodyClick)
73
74     this.input.removeEventListener( type: 'click', this.handleInputClick)
75 }
76
77 handleBodyClick = () => {
78     this.setState( state: { historyVisibility: false })
79 }
80
81 handleInputClick = (e: Event) => {
82     e.stopPropagation()
83 }
84
85 handleInputChange = (e: ChangeEvent<HTMLInputElement>) => {
86     this.setState( state: { query: e.target.value })
87 }
88
89 handleInputFocus = (e: FocusEvent<HTMLInputElement>) => {
90     this.setState( state: {
91         queryOnFocus: (e.target as HTMLInputElement).value,
92         historyVisibility: (this.props.history.length > 0),
93     })
94 }
95
96 handleInputBlur = () => {
97     this.props.onChange(this.input.value)
98
99     this.setState( state: {
100         query: this.input.value,
101         queryOnBlur: this.input.value,
102         historyVisibility: false,
103     })
104 }
```



```

19 class LoggedInput extends React.Component<LoggedInputProps, LoggedInputState> {
20   input: HTMLInputElement
21   historyBox: HTMLUListElement
22   state = {
23     query: '',
24     queryOnFocus: '',
25     queryOnBlur: '',
26     historyVisibility: false,
27     historyCursor: 0,
28   }
29
30   constructor(props: LoggedInputProps) {
31     super(props)
32
33     this.state.query = (this.props.defaultValue || []).join(' ')
34   }
35
36   get cursor(): number {
37     return this.state.historyCursor
38   }
39
40   set cursor(next: number) {
41     const index = next > this.props.history.length
42       ? this.props.history.length : next < 0
43       ? 0 : next
44
45     if (this.state.historyVisibility && index > 0) {
46       const item = this.historyBox.children[index - 1] as HTMLLIElement
47       this.historyBox.scrollTop = item.offsetTop + item.offsetHeight - this.historyBox.offsetHeight
48     } else {
49       this.historyBox.scrollTop = 0
50     }
51
52     this.props.onChange(next > 0
53       ? this.props.history[this.props.history.length - index]
54       : this.state.queryOnFocus)
55
56     this.setState({ state: {
57       query: next > 0
58         ? this.props.history[this.props.history.length - index]
59         : this.state.queryOnFocus,
60       historyVisibility: next > 0 && (this.props.history.length > 0),
61       historyCursor: index,
62     }})
63   }

```

