<http://blog.csdn.net/thystar/article/details/48047519>

http://www.runoob.com/jsref/dom-obj-event.html

事件；

1》触摸类事件：只会在移动设备中产生，对手的移动位置进行检测并做出响应

* onTouchCancel:
* onTouchEnd
* onTouchMove
* onTouchStart

2》键盘类事件：

* onKeyDown
* onKeyUp
* onKeyPress

3》剪切类事件

* onCopy
* onCut
* onPaste

4》 表单类事件

* onChange
* onInput
* onSubmit

5》 焦点类事件

* onFocus ： 获得焦点
* onBlur ： 失去焦点

6》UI元素： 元素或页面的滚动事件

* onScroll

7》滚动事件：监听滚动位置，方向

* onWheel

8》鼠标类事件：

* onClick
* onContextMenu ：右键，上下文菜单
* onDoubleClickc
* onMouseEnter
* onMouseDown
* onMouseLeave
* onMouseMove
* onMouseOut
* onMouseOver
* onMouseUp

9》鼠标拖拽事件： 上传内容

* onDrop
* onDrag
* onDragEnd
* onDragEnter
* onDragExit
* onDragLeave
* onDragOver
* onDragStart

实例1：

**[html]** [view plain](http://blog.csdn.net/thystar/article/details/48047519) [copy](http://blog.csdn.net/thystar/article/details/48047519)

1. <!DOCTYPE html>
2. <html>
3. <head lang="en">
4. <meta charset="UTF-8">
5. <title></title>
6. <script src="build/react.js"></script>
7. <script src="build/JSXTransformer.js"></script>
8. </head>
9. <body>
10. <script type="text/jsx">
11. var style = {
12. color : "red",
13. border: "1px #000 solid"
14. };
15. var HelloWorld = React.createClass({
16. handleChange: function (event) {
17. console.log(event.target.value);
18. },
19. render: function () {
20. return <div>
21. <input onChange={this.handleChange} />
22. </div>;
23. },
24. });
25. React.render(<div style={style}>
26. <HelloWorld></HelloWorld>
27. </div>, document.body);
28. </script>
29. </body>
30. </html>

2 事件对象：

**[html]** [view plain](http://blog.csdn.net/thystar/article/details/48047519) [copy](http://blog.csdn.net/thystar/article/details/48047519)

1. event.target.value

target：事件对象的属性， 事件应得DOM元素

事件对象的属性：

1》通用属性

* boolean  bubbles : 事件是否可以冒泡
* boolean cancelable : 事件是否可以取消
* DOMEventTarget currentTarget :
* boolean defaultPrevented : 事件是否禁止默认行为
* number eventPhase : 事件所处的阶段
* boolean isTrusted : 事件是否可信，可信事件即用户自定义触发的事件，不可信事件即用js代码触发的事件
* DOMEvent nativeEvent  使用原生的浏览器发出的事件对象
* void preventDefault() : 禁止默认行为
* void stopPropagation() ： 禁止冒泡
* DOMEventTarget target:
* number timeStamp
* string type

2》不同事件对象特有的属性

i  剪切事件：

* DOMDataTransfer clipboardDate : 得到剪切数据

ii 键盘事件：

* boolean altKey ; 是否按下alt键
* Number charCode ；按键的编码; 字符编码
* boolean ctrlKey ；是否按下ctrl键
* function getModifierState(key) ; 是否按下辅助按键ctrl,shift
* String key
* Number keyCode; 按键编码；非字符
* String locale ; 本地化的字符串
* Number location ；位置
* boolean metaKey; win键
* boolean repeat ；按键是否重复
* boolean shiftKey； 是否按下shift
* Number which ; 通用化的charCode和keyCode

iii 焦点

* DOMEventTarget relatedTarget :  相关的角度

iv 鼠标事件

* boolean altKey；
* Number button；
* Number buttons；
* Number clientX;
* Number clientY; 当前鼠标所处的坐标, 顶点是浏览器窗口的左上角
* boolean ctrlKey
* function getModifierState(key);
* boolean metaKey;
* Number pageX
* Number pageY; 顶点时html页面的左上角
* DOMEventTarget relatedTarget ;
* Number screenX；
* Number scrrenY; p
* boolean shiftKey;

v 触摸事件：

* boolean altKey
* DOMTouchList changedTouchs
* boolean ctrlKey
* function getModifierState(Key)
* boolean metaKey
* boolean shiftKey
* DOMTouchList targetTouches
* DOMTouchList touches

vi UI元素

* Number detail ： 滚动的距离
* DOMAbstractView view ： 视图

vii 滚动：

* Number deltaMode: 单位
* Number deltaX
* Number deltaY
* Number deltaZ 在坐标轴上对应的位置

实例1

**[html]** [view plain](http://blog.csdn.net/thystar/article/details/48047519) [copy](http://blog.csdn.net/thystar/article/details/48047519)

1. <!DOCTYPE html>
2. <html>
3. <head lang="en">
4. <meta charset="UTF-8">
5. <title></title>
6. <script src="build/react.js"></script>
7. <script src="build/JSXTransformer.js"></script>
8. </head>
9. <body>
10. <script type="text/jsx">
11. var HelloWorld = React.createClass({
12. getInitialState: function () {
13. return{
14. backgroundColor:'#FFFFFF'
15. }
16. },
17. handleWheel: function (event) {
18. var newColor=(parseInt(this.state.backgroundColor.substr(1),16) +
19. event.deltaY \* 997).toString(16);
20. newColor = '#' + newColor.substr(newColor.length - 6).toUpperCase();
21. this.setState({
22. backgroundColor: newColor
23. });
24. },
25. render: function () {
26. console.log(this.state);
27. return <div onWheel={this.handleWheel} style={this.state}>
28. <p>Hello World</p>
29. </div>;
30. },
31. });
32. React.render(
33. <HelloWorld></HelloWorld>
34. , document.body);
35. </script>
36. </body>
37. </html>

实例2

**[html]** [view plain](http://blog.csdn.net/thystar/article/details/48047519) [copy](http://blog.csdn.net/thystar/article/details/48047519)

1. <!DOCTYPE html>
2. <html>
3. <head lang="en">
4. <meta charset="UTF-8">
5. <title></title>
6. <script src="build/react.js"></script>
7. <script src="build/JSXTransformer.js"></script>
8. </head>
9. <body>
10. <script type="text/jsx">
11. var HelloWorld = React.createClass({
12. getInitialState: function () {
13. return {
14. password: ''
15. }
16. },
17. handleKeyPress: function (event) {
18. this.setState({
19. password: this.state.password + event.which
20. });
21. console.log(this.state)
22. },
23. handleChange: function (event) {
24. event.target.value = '';
25. },
26. render: function () {
27. return <div>
28. <input onKeyPress={this.handleKeyPress} onChange={this.handleChange}/>
29. <p style={{'display':this.state.password.indexOf('495051')>=0 ? 'inline'
30. :'none'
31. }}>You got it</p>
32. </div>
33. },
34. });
35. React.render(
36. <HelloWorld></HelloWorld>
37. , document.body);
38. </script>
39. </body>
40. </html>

3  事件和状态关联

this.setState()

实例1：

**[html]** [view plain](http://blog.csdn.net/thystar/article/details/48047519) [copy](http://blog.csdn.net/thystar/article/details/48047519)

1. <!DOCTYPE html>
2. <html>
3. <head lang="en">
4. <meta charset="UTF-8">
5. <title></title>
6. <script src="build/react.js"></script>
7. <script src="build/JSXTransformer.js"></script>
8. </head>
9. <body>
10. <script type="text/jsx">
11. var HelloWorld = React.createClass({
12. getInitialState: function () {
13. return {
14. x:0,
15. y:0
16. }
17. },
18. handleMouseMove: function (event) {
19. this.setState({
20. x:event.clientX,
21. y:event.clientY
22. })
23. },
24. render: function () {
25. return <div onMouseMove={this.handleMouseMove} style={{
26. height:'1000px',
27. width:'700px',
28. backgroundColor: 'gray'
29. }}>
30. {this.state.x + ', ' + this.state.y}
31. </div>
32. },
33. });
34. React.render(
35. <HelloWorld></HelloWorld>
36. , document.body);
37. </script>
38. </body>
39. </html>

视频课程： 极客学院