React SyntheticEvent

2023.03.08 민경민

왜 SyntheticEvent 가 필요한가?

- Cross Browser Support
- Performance Optimization

Browser Event

DOM node가 생성한 시그널

Mouse events:

- `click` – when the mouse clicks on an element (touchscreen devices generate it on a tap).
- `contextmenu` – when the mouse right-clicks on an element.
- `mouseover` / `mouseout` – when the mouse cursor comes over / leaves an element.
- `mousedown` / `mouseup` – when the mouse button is pressed / released over an element.
- `mousemove` – when the mouse is moved.

Keyboard events:

- `keydown` and `keyup` – when a keyboard key is pressed and released.

Form element events:

- `submit` – when the visitor submits a `<form>`.
- `focus` – when the visitor focuses on an element, e.g. on an `<input>`.

Document events:

- `DOMContentLoaded` – when the HTML is loaded and processed, DOM is fully built.

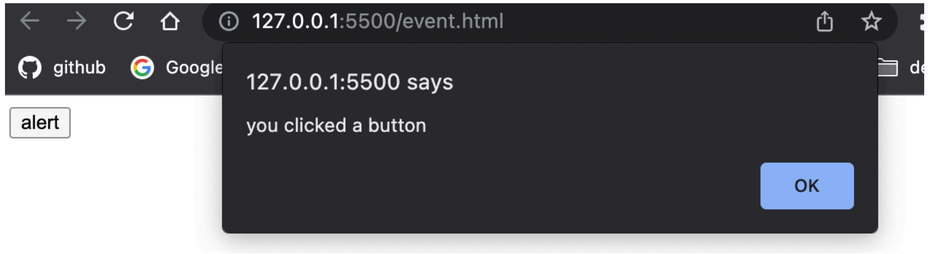
CSS events:

- `transitionend` – when a CSS-animation finishes.

Event Handler

Event 발생 시 실행 되는 함수

```
<body>
  <button onclick="alert('you clicked a button')">alert</button>
</body>
```




```
<button onclick="handleClick()">alert multiple</button>
<script>
  function handleClick() {
    for (let i = 0; i <= 3; i++) {
      alert("marry had a little lamb");
    }
  }
</script>
```

this로 DOM node에 접근 할 수 있다

```
<button onclick="alert(this.innerText)">
  You can access element via this
</button>
```

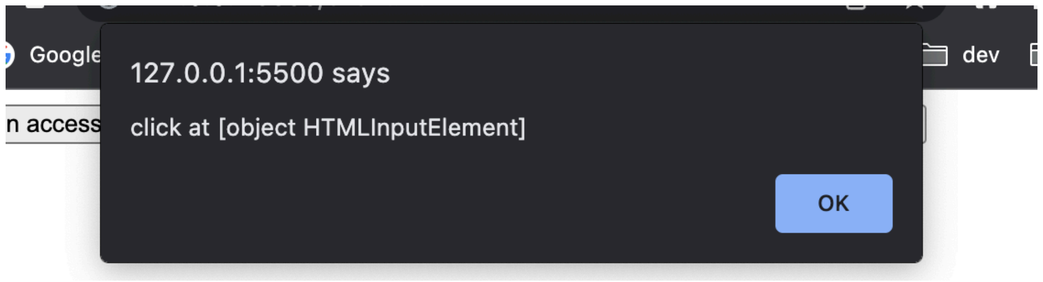
```
<button onclick="this.remove()">remove me</button>
```

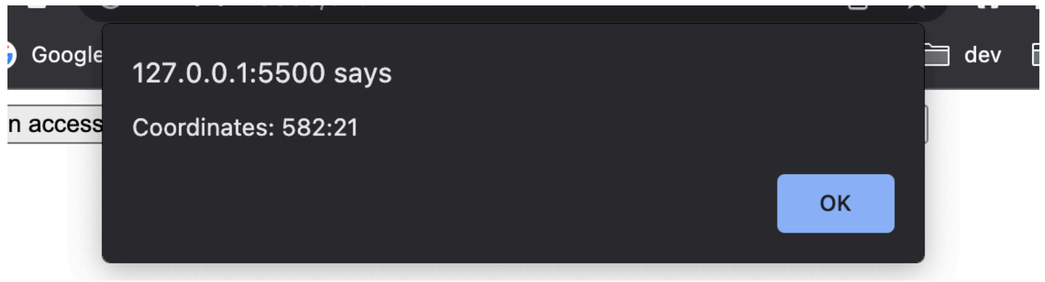
```
<button id="apple-sauce">apple sauce</button>
<script>
  const button = document.querySelector("#apple-sauce");
  button.onclick = function () {
    alert(this.innerText);
  };
  // or
  // button.addEventListener("click", function () {
  //   this.remove();
  // });
</script>
```

Event Object

handler에게 전달되는 argument

```
<script>
  elem.onclick = function (event) {
    // show event type, element and coordinates of the click
    alert(event.type + " at " + event.currentTarget);
    alert("Coordinates: " + event.clientX + ":" + event.clientY);
  };
</script>
```





Cross Browser 서포트는 귀찮다...

```
/**
 * Cross Browser helper to addEventListener.
 *
 * @param {HTMLElement} obj The Element to attach event to.
 * @param {string} evt The event that will trigger the binded function.
 * @param {function(event)} fnc The function to bind to the element.
 * @return {boolean} true if it was successfuly binded.
 */
var cb_addEventListener = function(obj, evt, fnc) {
    // W3C model
    if (obj.addEventListener) {
        obj.addEventListener(evt, fnc, false);
        return true;
    }
    // Microsoft model
    else if (obj.attachEvent) {
        return obj.attachEvent('on' + evt, fnc);
    }
    // Browser don't support W3C or MSFT model, go on with traditional
    else {
        evt = 'on'+evt;
        if(typeof obj[evt] === 'function'){
            // Object already has a function on traditional
            // Let's wrap it with our own function inside another function
            fnc = (function(f1,f2){
                return function(){
                    f1.apply(this,arguments);
                    f2.apply(this,arguments);
                }
            })(obj[evt], fnc);
        }
        obj[evt] = fnc;
        return true;
    }
    return false;
};
```

Thus SyntheticEvents

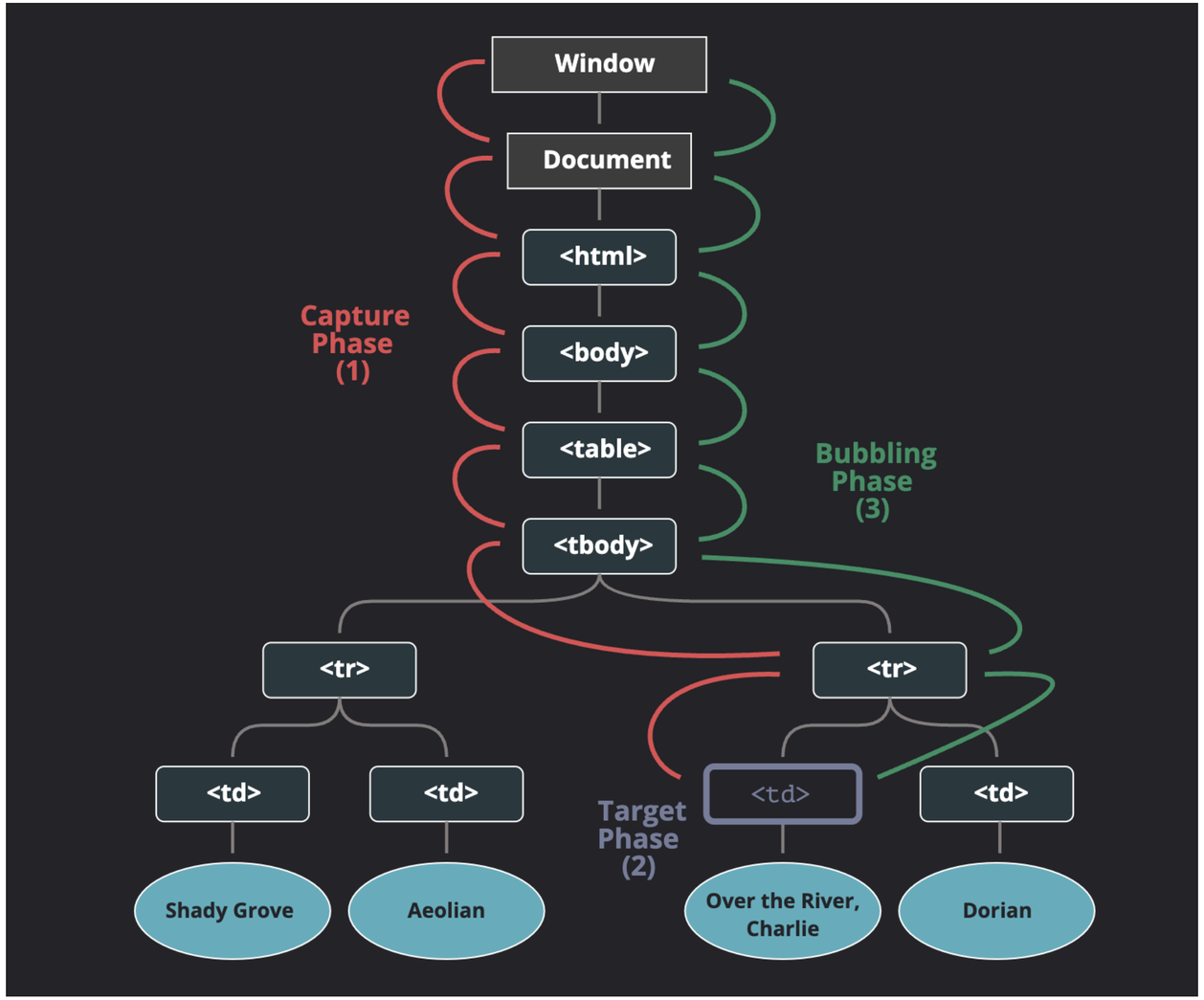
Overview

Your event handlers will be passed instances of `SyntheticEvent`, a cross-browser wrapper around the browser's native event. It has the same interface as the browser's native event, including `stopPropagation()` and `preventDefault()`, except the events work identically across all browsers.

어떻게 구현 했을까?

However, for most events, React doesn't actually attach them to the DOM nodes on which you declare them. Instead, React attaches one handler per event type directly at the `document` node. This is called event delegation. In addition to its performance benefits on large application trees, it also makes it easier to add new features like replaying events.

Event Propagation



Bubbling, Capturing, Target

- Capturing Phase : event가 target element로 top-down으로 이동한다
- Target Phase : event가 target element에 도착
- Bubbling Phase : event가 최상위 부모까지 bottom-up으로 이동한다

event.target, event.currentTarget

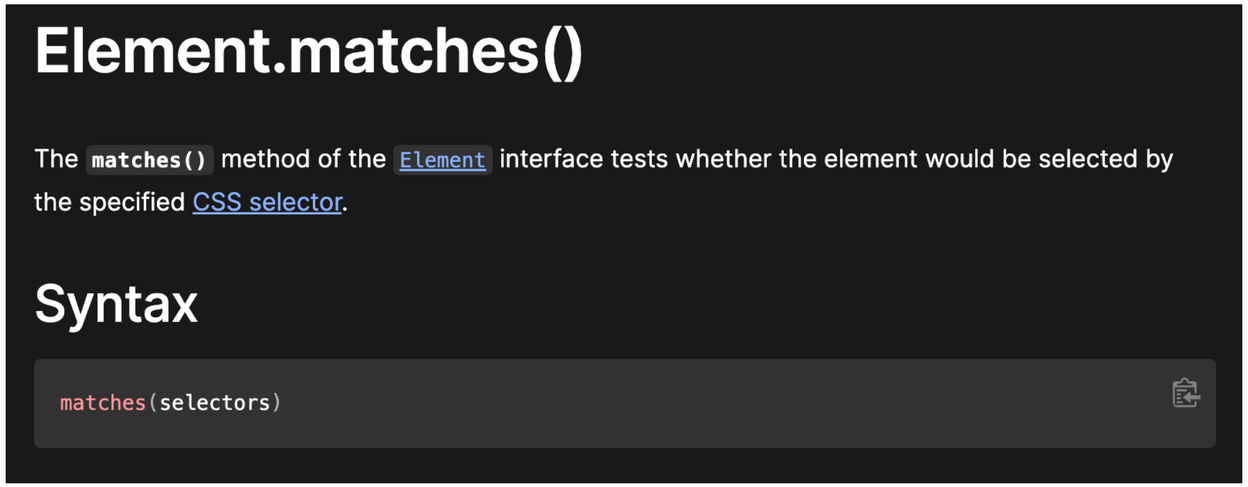
- event.target은 event를 발생시킨 element
- event.currnetTarget은 handler를 실행 시키고 있는 element (this)

```
1  form.onclick = function(event) {
2    event.target.style.backgroundColor = 'yellow';
3
4    // chrome needs some time to paint yellow
5    setTimeout(() => {
6      alert("target = " + event.target.tagName + ", this=" + this.tagName);
7      event.target.style.backgroundColor = ''
8    }, 0);
9  };
```

Event Delegation

```html
<ul id="parent-list">
    <li id="post-1">Item 1</li>
    <li id="post-2">Item 2</li>
    <li id="post-3">Item 3</li>
    <li id="post-4">Item 4</li>
    <li id="post-5">Item 5</li>
    <li id="post-6">Item 6</li>
</ul>
```

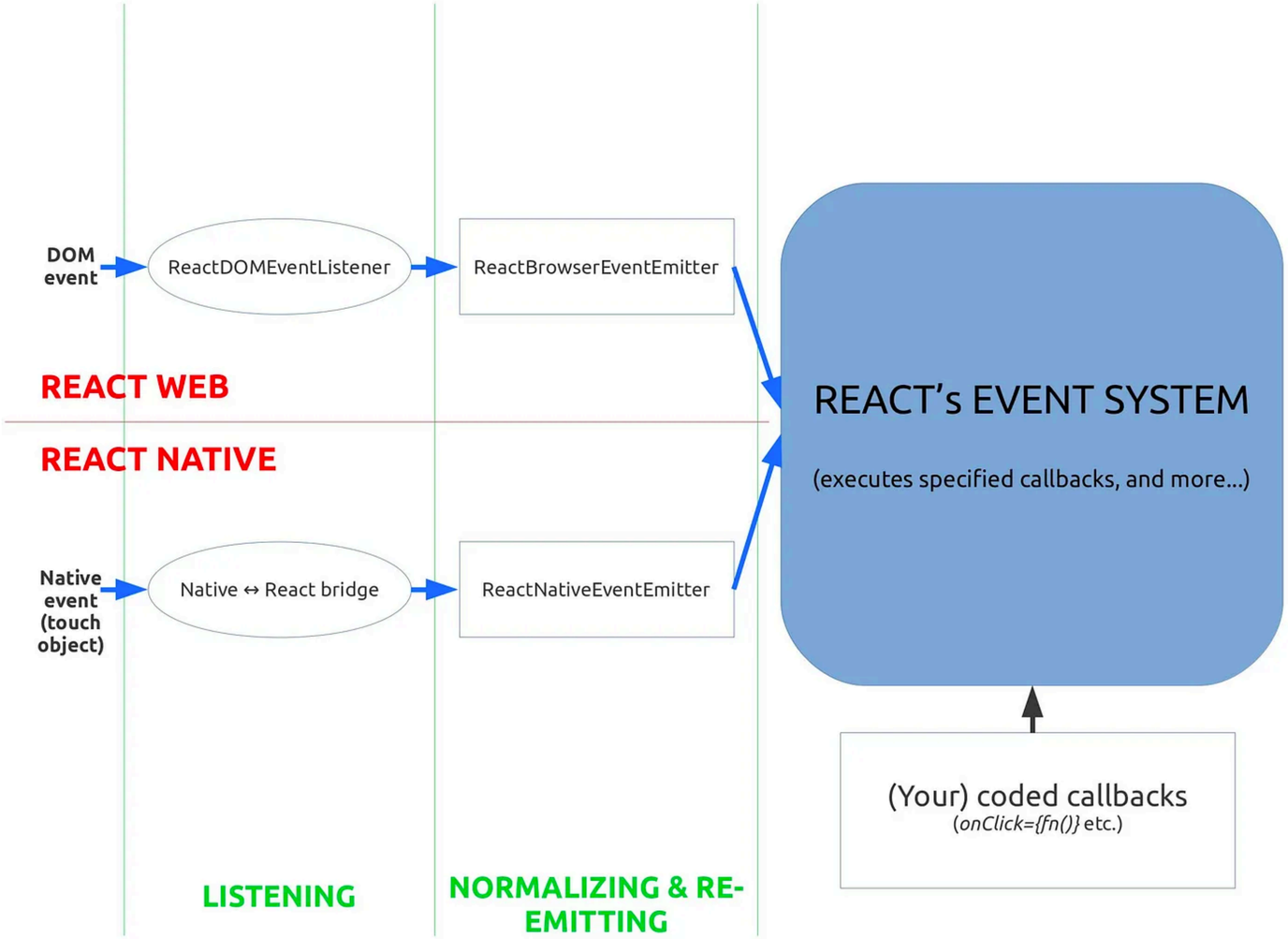
```js
// Get the element, add a click listener...
document.getElementById("parent-list").addEventListener("click", function(e) {
    // e.target is the clicked element!
    // If it was a list item
    if(e.target && e.target.nodeName == "LI") {
        // List item found!  Output the ID!
        console.log("List item ", e.target.id.replace("post-", ""), " was clicked!");
    }
});
```



Event Delegation

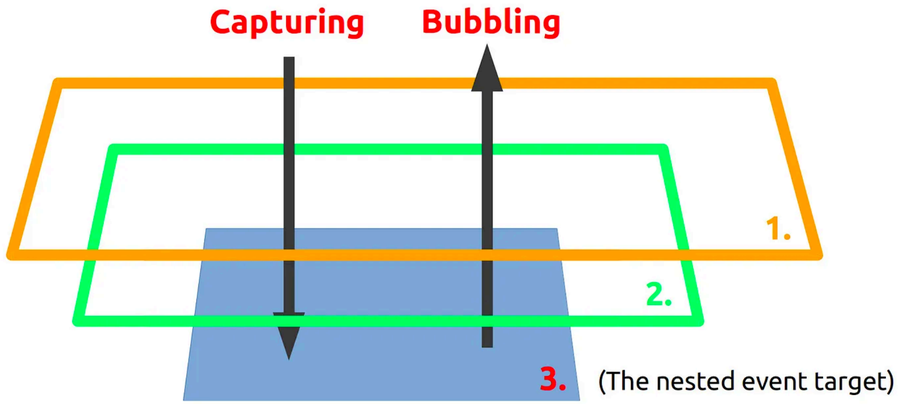
- Event Delegation는 부모에게 handler를 붙이고 event.target이 어떤 element인지 체크하여 event를 처리하는 기법이다
- Event Delegation을 사용하면 element 변경에 일일히 대응하지 않아도 된다
- Naive 하게 생각한다면 document element에 switch case를 포함한 handler를 등록한다면 모든 event를 처리할 수 있다

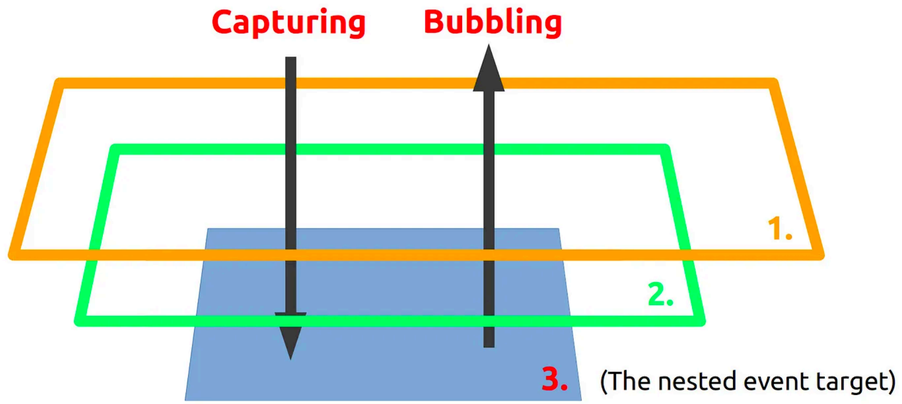
React Event System



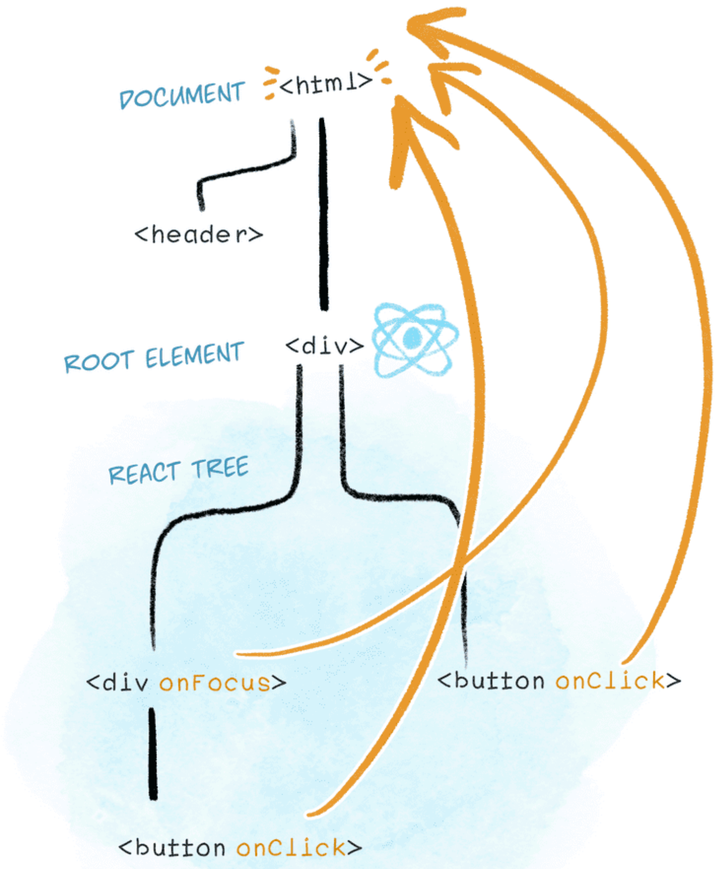
Event & Synthetic Event

Traversal이 두번 일어난다

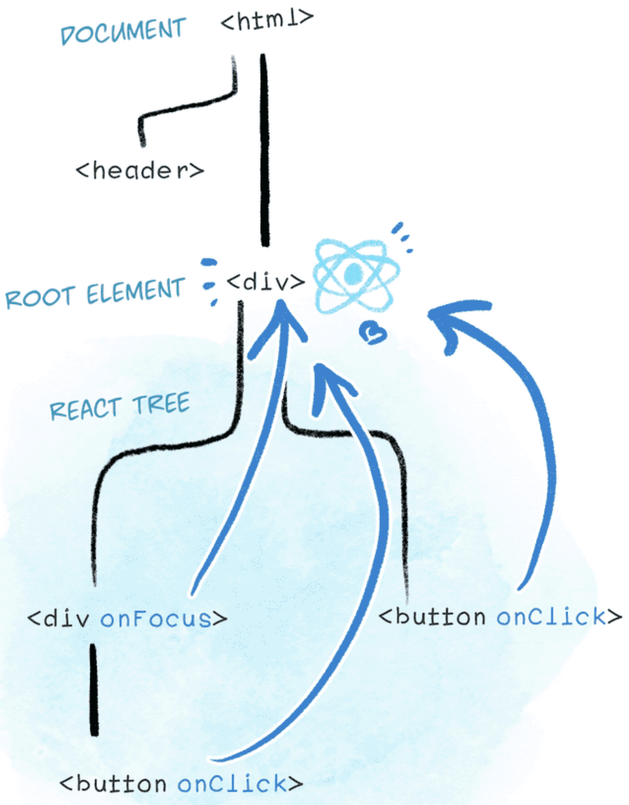




REACT 16



REACT 17



@RachelNabors

BaseSyntheticEvent

```
--
23  type BaseSyntheticEvent = {
24    isPersistent: () => boolean,
25    isPropagationStopped: () => boolean,
26    _dispatchInstances?: null | Array<Fiber | null> | Fiber,
27    _dispatchListeners?: null | Array<Function> | Function,
28    _targetInst: Fiber,
29    nativeEvent: Event,
30    target?: mixed,
31    relatedTarget?: mixed,
32    type: string,
33    currentTarget: null | EventTarget,
34  };
```

https://github.com/facebook/react/blob/62cd5af08e2ac8b1d4691e75252487083cf7a4aa/packages/react-dom-bindings/src/events/ReactSyntheticEventType.js#L23

참고

- https://reactjs.org/docs/events.html
- https://reactjs.org/blog/2020/08/10/react-v17-rc.html#fixing-potential-issues
- https://davidwalsh.name/event-delegate
- https://javascript.info/bubbling-and-capturing