Swipe effect in react js

Asked 8 years, 1 month ago Modified 1 year, 2 months ago Viewed 86k times

30

I'm trying to create a swipe event using React. I do not want to use any external component or jquery.

The css is something like :

```
.outter{
  position:relative;
  width: 100%;
  height: 150px;
  background-color: blue;
}

.inner{
  position: absolute;
  width: 1000%;
  left: 50px;
}
.child{
  float: left;
  margin-right: 15px;
}
```

In the react component I'm trying to do something like :

```
class Test extends React.Component {
    constructor(props){
    super(props);

    this.state = {
        left: 0
    }
  }

  handleSwipe(){
    this.setState({left: -350})
  }

  render(){
    return(
        <div className="outter">
            <div className="inner" style={{left: this.state.left}} onSwipe=
{this.handleSwipe.bind(this)}>
                <div className="child"><img src="http://placehold.it/350x150"
/></div>
                <div className="child"><img src="http://placehold.it/350x150"
/></div>
                <div className="child"><img src="http://placehold.it/350x150"
/></div>
                <div className="child"><img src="http://placehold.it/350x150"
/></div>
```

```
            </div>
        </div>
      )
    }
}

React.render(<Test />, document.getElementById('container'));
```

How can I recognize swipe event?

If I in my example instead of `onSwipe` add `onClick` it works, but how can I make the swipe effect?

Here is jsfiddle.

javascript reactjs

Share
Improve this question
Follow

edited Feb 13, 2018 at 8:11

asked Nov 7, 2016 at 10:40

Boky
12.1k ● 30 ● 101 ● 171

3 If you really want to do it yourself, then you need to listen and handle mousedown + mousemove + mouseup events. – dfsq Nov 7, 2016 at 10:42

4 Answers

Sorted by: Highest score (default)

64

You can add `onTouch` event handlers to your React components:

```
onTouchStart={touchStartEvent => this.handleTouchStart(touchStartEvent)}
onTouchMove={touchMoveEvent => this.handleTouchMove(touchMoveEvent)}
onTouchEnd={() => this.handleTouchEnd()}
```

You may also want to add event handlers for mouse events for cross-platform compatibility:

```
onMouseDown={mouseDownEvent => this.handleMouseDown(mouseDownEvent)}
onMouseMove={mouseMoveEvent => this.handleMouseMove(mouseMoveEvent)}
onMouseUp={() => this.handleMouseUp()}
onMouseLeave={() => this.handleMouseLeave()}
```

You have the right idea for updating the `left` property of state in an event handler, but if you want the swiping functionality to feel natural you'll need to track the position

of the pointer (be it a mouse or a touch) by updating `left` using the event's `clientX` property.

To do this, you'll need to store the position of the first touch and set `left` equal to the change in the touch's location. For added realism, you can also keep track of the touch's velocity and continue animating the component after the touch is finished.

Here's a quick-n-dirty Codepen I made for swiping to remove items from a list:

https://codepen.io/swingthing/pen/ZBGBJb/

Share

Improve this answer

Follow

edited Nov 4, 2019 at 9:10

Yashwardhan Pauranik

5,536 ● 5 ● 44 ● 74

answered Nov 7, 2016 at 22:14

Jake Hoffman

686 ● 6 ● 6

13 Just a note: `onTouchStart={touchStartEvent => this.handleTouchStart(touchStartEvent)}` is functionally equivalent to `onTouchStart={this.handleTouchStart}` but is somewhat less efficient, as it causes a new anonymous function to be defined every render. – chadoh Dec 30, 2017 at 16:06

1 Great answer and codepen. Strangely, when copying you, I end up with a much more laggy experience. – chadoh Dec 30, 2017 at 16:10

2 I wish everyones quick and dirty code read like yours sir. 👏👏👏 – Brandon Howard Feb 11, 2018 at 3:40

So basically onSwipe event does not exist for React, only for React-Native, we use onTouch to replace onSwipe. – artworkjpm Feb 24, 2021 at 14:18

1 @jake Hoffman Firstly, thank you for this great piece of code art. I was wondering if you have code for handleMotionStart function i am unable to find its code in above codepen. Thanks in advance – Gaurav Apr 22, 2021 at 0:44

30

I used solution from most popular answer as a foundation for build what I was needed. Maybe somebody need something like this.

Idea was to move my custom slider left and right on swipe effect. If you want it to be more sensitive - adjust value 150 to 75.

```
const [touchStart, setTouchStart] = React.useState(0);
const [touchEnd, setTouchEnd] = React.useState(0);

function handleTouchStart(e) {
    setTouchStart(e.targetTouches[0].clientX);
}

function handleTouchMove(e) {
    setTouchEnd(e.targetTouches[0].clientX);
}
```

```
function handleTouchEnd() {
    if (touchStart - touchEnd > 150) {
        // do your stuff here for left swipe
        moveSliderRight();
    }

    if (touchStart - touchEnd < -150) {
        // do your stuff here for right swipe
        moveSliderLeft();
    }
}
```

Share

Improve this answer

Follow

edited Oct 11, 2023 at 21:13

answered Oct 19, 2020 at 17:25

Nik B

647 ● 7 ● 11

6

Might be for someone my solution will be also useful:

```
export const useSwipe = ({left, right, up, down}: {left?: () => void; right?:
() => void; up?: () => void; down?: () => void;}) => {
  const touchCoordsRef = useRef({touchStart: {x: 0, y: 0, time: Date.now()}});
  const fnsRef = useRef({up, down, left, right});
  fnsRef.current = {
    up,
    left,
    down,
    right,
  };
  useEffect(() => {
    const handleTouchStart = (e: TouchEvent) => {
      touchCoordsRef.current.touchStart.x = e.targetTouches[0].clientX;
      touchCoordsRef.current.touchStart.y = e.targetTouches[0].clientY;
      touchCoordsRef.current.touchStart.time = Date.now();
    };
    const handleTouchEnd = (e: TouchEvent) => {
      const threshold = 150;
      const swipeSpeed = 1; // sec;
      const touchEndX = e.changedTouches[0].clientX;
      const touchEndY = e.changedTouches[0].clientY;
      const touchStartX = touchCoordsRef.current.touchStart.x;
      const touchStartY = touchCoordsRef.current.touchStart.y;
      const elapsedTime = (Date.now() - touchCoordsRef.current.touchStart.time)
/ 1000;
      if(elapsedTime > swipeSpeed) {
        return;
      }
      const xDistance = touchStartX - touchEndX;
      const yDistance = touchStartY - touchEndY;

      if(Math.abs(xDistance) < threshold && Math.abs(yDistance) < threshold) {
        return;
      }

      if(Math.abs(xDistance) >= Math.abs(yDistance)) {
        xDistance > 0 ? fnsRef.current.right?.() : fnsRef.current.left?.();
```

```
      } else {
        yDistance > 0 ? fnsRef.current.down?.() : fnsRef.current.up?.();
      }
    };
    window.addEventListener('touchstart', handleTouchStart);
    window.addEventListener('touchend', handleTouchEnd);
    return () => {
      window.removeEventListener('touchstart', handleTouchStart);
      window.removeEventListener('touchend', handleTouchEnd);
    };
  });
};
```

Share Improve this answer Follow

answered Jun 9, 2022 at 23:14

Denis Khay
3,870 ● 1 ● 11 ● 6

Very nice, although I think the invoking of `right()` and `left()` should be reversed. When moving from the right of the screen to the left, `xDistance` will be positive, and I'd call that a left swipe (pushing the screen left). I think swiping up/down is correct here though – devklick Jun 19, 2023 at 13:21

-5

Just found a DOM event that could be useful. Add .ontouchend to your HTML tag to detect any swipe event originating from that tag.

https://www.w3schools.com/jsref/event_touchend.asp - ref

Share Improve this answer Follow

answered Nov 19, 2020 at 16:16

Jeremiah
1