Автор Марочкин М.В.

1. Определенные во фреймворке события **tap**, **up**, **down** и **track** работают только в режиме всплытия. При назначении методом **listen** обработчика для этих событий значение параметра **useCapture игнорируется.**
2. Для этих событий, при отключении обработчика методом **unlisten** параметр **useCapture** также игнорируется**.**

Поэтому возникает вопрос, что делать? Можно прямо указать на эту особенность в документации, или доработать фреймворк, чтобы указанные события работали и на всплытие и на погружение.

Пример для демонстрации первой проблемы:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-component',

template: `

<div ref='div2up' style='background:lightgreen; width:500px; height:90px'>

<div ref='div1up' style='background:red; width:300px; height:50px'>

<button ref='btnUP'>Всплывающее событие</button>

</div>

</div>

<hr>

<div ref='div2down' style='background:lightblue; width:500px; height:90px'>

<div ref='div1down' style='background:pink; width:300px; height:50px'>

<button ref='btnDOWN'>Погружающееся событие</button>

</div>

</div>

`,

props: {

event: 'click',

// event: 'tap',

// event: 'up',

// event: 'down'

},

ready() {

this.listen(this.event, ()=>console.log(this.event + '---button UP'), {target: this.$refs.btnUP, useCapture: false});

this.listen(this.event, ()=>console.log(this.event + '-------div1 UP'), {target: this.$refs.div1up, useCapture: false});

this.listen(this.event, ()=>console.log(this.event + '-----------div2 UP'), {target: this.$refs.div2up, useCapture: false});

this.listen(this.event, ()=>console.log(this.event + '===button DOWN'), {target: this.$refs.btnDOWN, useCapture: true});

this.listen(this.event, ()=>console.log(this.event + '=======div1 DOWN'), {target: this.$refs.div1down, useCapture: true});

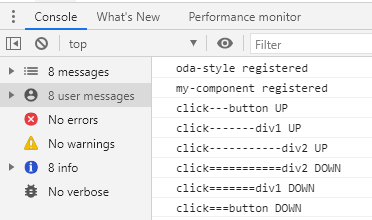
this.listen(this.event, ()=>console.log(this.event + '===========div2 DOWN'), {target: this.$refs.div2down, useCapture: true});

});

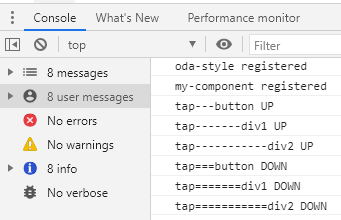
</script>

В примере использованы две группы HTML-элементов. В верхней группе события должны всплывать, а в нижней – погружаться. Для контроля порядка срабатывания событий каждый элемент выводит в консоль соответствующее сообщение.

При регистрации нативных событий (например, события **click** как в примере выше) фреймворк правильно отрабатывает направление распространения событий. Это можно видеть по сообщениям в консоли:



При использовании события **tap**, оно будет только всплывать от кнопки:



Пример для демонстрации второй проблемы:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-component',

template: `

<button ref='btnUP'>Всплывающее событие</button>

<button ref='btnDOWN'>Погружающееся событие</button>

<div>countUP={{countUP}}</div>

<div>countDown={{countDown}}</div>

`,

props: {

event: 'click',

// event: 'tap',

// event: 'up',

// event: 'down',

countUP: 0,

countDown: 0

},

up() {

this.countUP++;

this.unlisten(this.event, 'up', {target: this.$refs.btnUP, useCapture: true});

},

down() {

this.countDown++;

this.unlisten(this.event, 'down', {target: this.$refs.btnDOWN, useCapture: false});

},

ready() {

this.listen(this.event, 'up', {target: this.$refs.btnUP, useCapture: false});

this.listen(this.event, 'down', {target: this.$refs.btnDOWN, useCapture: true});

}

});

</script>

В этом примере для двух кнопок методом **listen** регистрируются два обработчика. Один обработчик регистрируется на всплытие, второй – на погружение. В коде обработчиков их регистрация отменяется методом **unlisten**, но отменяется для другой фазы распространения события (значения параметра **useCapture** в обоих методах не совпадают), т.е. фактически зарегистрированные обработчики должны продолжать вызываться при наступлении события.

Для события **click** обработчик не отключается, как и ожидалось. Это видно по продолжающемуся инкременту счетчиков при нажатии на кнопки.

А для события **tap** счетчики инкрементируются только один раз, т.к. метод **unlisten** отключает обработчик не обращая внимания на несоответствие значения параметра **useCapture** зарегистрированному значению.