Автор Марочкин М.В.

1) Изменилась работа директивы **~save-key**, поэтому видео урок не соответствует текстовому описанию.

2) Если компонент имеет два свойства с модификатором **save**, но отсутствует директива **~save-key**, то сохраняется только первое по списку свойство. Точнее второе свойство тоже сохраняется, но только на одну сессию. Если перезагрузить страницу два раза, то после второго раза во втором свойстве будет значение, заданное по умолчанию. Тоже самое с перезагрузкой браузера. После первой перезагрузки второе свойство будет иметь сохраненное значение, а после второй перезагрузки второе свойство будет иметь значение по умолчанию, заданное в компоненте.

Пример для демонстрации проблемы:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-input-component',

template: `

<input ::value="value1">

<input ::value="value2">

`,

props: {

value1: {

default: 'Введите текст 1',

save: true

},

value2: {

default: 'Введите текст 2',

save: true

},

}

});

ODA({

is: 'my-component',

template: `

<my-input-component></my-input-component>

`

});

</script>

3) Не работает очистка хранилища функцией **clearSaves()**, если одновременно происходит процесс рендеринга страницы, например, при изменении одного из свойств компонента.

Пример для демонстрации проблемы, хранилище очищается только на 4-е нажатие кнопки **Reset** после открытия страницы:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-input-component',

template: `

<input ::value="value1">

<button @tap="clearMyStorage()">Reset</button>

`,

props: {

value1: {

default: 'Введите текст',

save: true

},

},

clearMyStorage() {

if( this.domHost.counter < 3 )

this.domHost.counter++;

this.clearSaves();

},

});

ODA({

is: 'my-component',

template: `

<my-input-component ~save-key="'myKey'"></my-input-component>

<span>Количество нажатий: {{counter}}</span>

`,

props: {

counter: 0

}

});

</script>

4) Не работает упрощенный вариант директивы **save-key** без тильды. В примере видно, что директива создается с «пустым» ключом:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-input-component',

template: `

<input ::value="value1">

`,

props: {

value1: {

default: 'Введите текст',

save: true

},

},

});

ODA({

is: 'my-component',

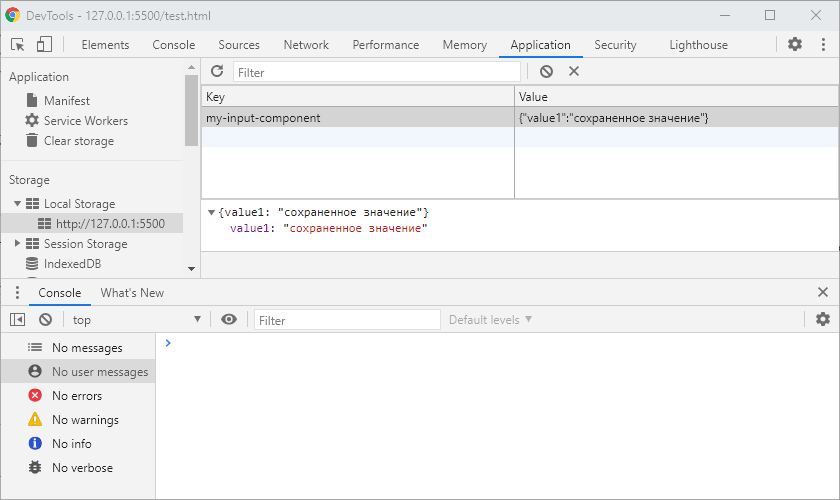
template: `

<my-input-component save-key="myKey"></my-input-component>

`,

});

</script>



5) Если компонент имеет два свойства с модификатором **save**, но отсутствует директива **~save-key**, т.е. данные сохраняются с «пустым» ключом, то сохраняется только первое по списку свойство. Точнее второе свойство тоже сохраняется, но только на одну сессию. Если перезагрузить страницу два раза, то после второго раза во втором свойстве будет значение, заданное по умолчанию.

То же самое с перезагрузкой браузера. После первой перезагрузки второе свойство будет иметь сохраненное значение, а после второй перезагрузки второе свойство вернется к значению по умолчанию, заданному в компоненте.

Пример для демонстрации проблемы:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-input-component',

template: `

<input ::value="value1">

<input ::value="value2">

`,

props: {

value1: {

default: 'Введите текст 1',

save: true

},

value2: {

default: 'Введите текст 2',

save: true

},

},

});

ODA({

is: 'my-component',

template: `

<my-input-component></my-input-component>

`,

});

</script>

6) Если сохраняемый текст совпадает с текстом, указанным в компоненте по умолчанию, то такой текст не сохраняется. В хранилище остается предыдущее значение, которое загружается после перезагрузки страницы. Таким образом получается, что невозможно сохранить текст, совпадающий с используемым по умолчанию, и чтобы вернуться к значению по умолчанию необходимо очистить хранилище.

Пример для демонстрации проблемы:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-input-component',

template: `

<input ::value="value1">

`,

props: {

value1: {

default: 'Введите текст',

save: true

},

},

});

ODA({

is: 'my-component',

template: `

<my-input-component ~save-key="'myKey'"></my-input-component>

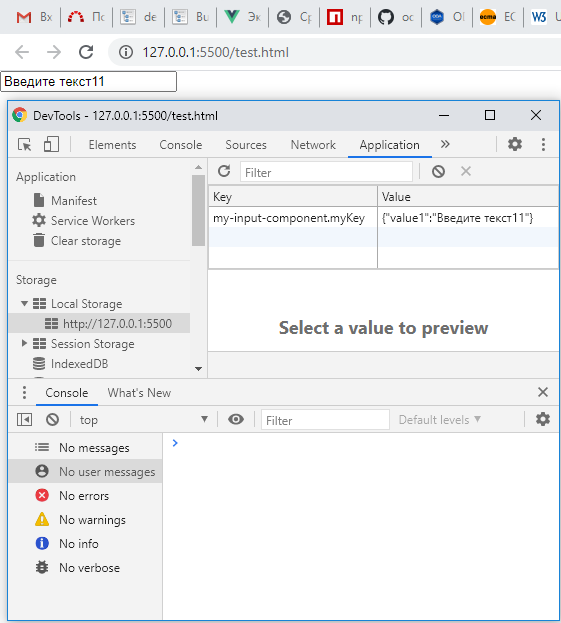
`,

});

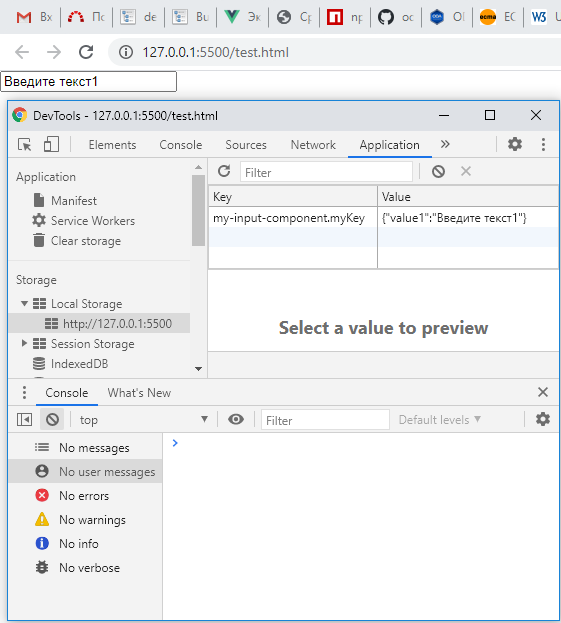
</script>

Изначально хранилище было пустое, и в строке ввода отобразился текст по умолчанию **«Введите текст11»**.

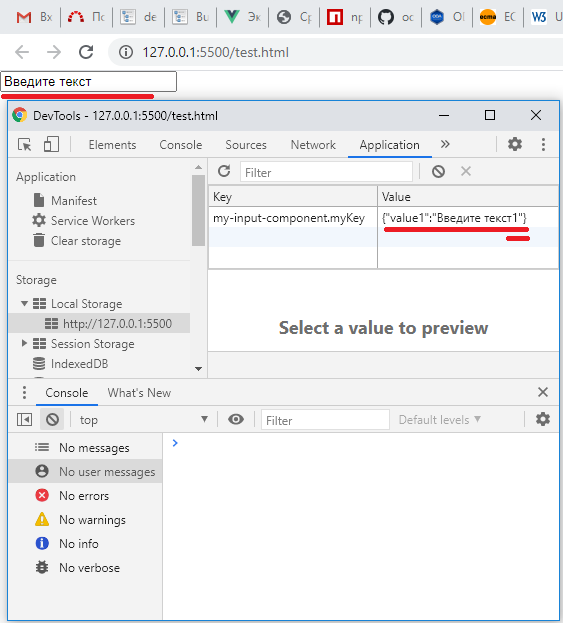
В примере к значению по умолчанию я добавил две 1 **«Введите текст11»**. Можно видеть, что в хранилище все сохранилось:



Удалил одну единицу, в хранилище все сохранилось:



Удалил вторую единицу. Теперь текст в строке ввода совпал со значением по умолчанию. Но можно видеть, что в хранилище осталось предыдущее значение (т.е. текст по умолчанию в хранилище не сохраняется), и именно это предыдущее значение восстанавливается при перезагрузке страницы:



Таким образом получается, что невозможно ввести и сохранить текст, совпадающий с используемым по умолчанию, и чтобы вернуться к значению по умолчанию необходимо очистить хранилище.

Пример с двумя сохраняемыми свойствам:

<meta charset="UTF-8">

<title>WELCOME</title>

<my-component></my-component>

<script type="module">

import "/oda.js";

ODA({

is: 'my-input-component',

template: `

<input ::value="value1">

<input ::value="value2">

`,

props: {

value1: {

default: 'Введите текст 1',

save: true

},

value2: {

default: 'Введите текст 2',

save: true

},

},

});

ODA({

is: 'my-component',

template: `

<my-input-component ~save-key="'myKey'"></my-input-component>

`,

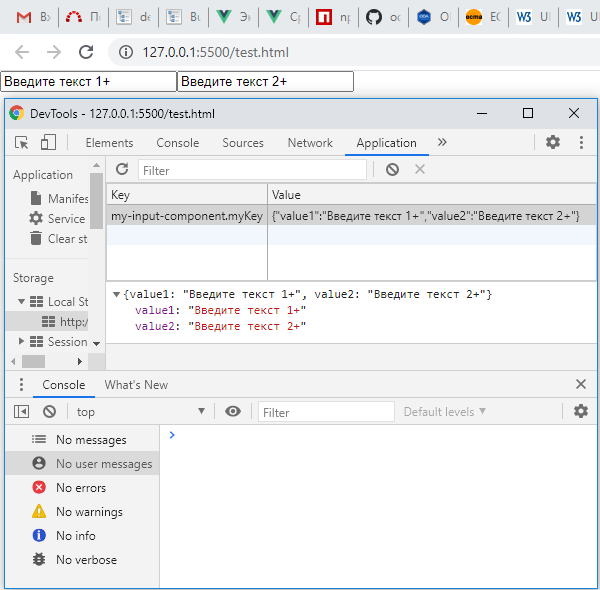
});

</script>

Пока во второе поле ввода не ввели новой записи, сохранение данных в первом поле идет с ошибкой описанной выше, т.е. невозможно вернуть первое поле к значению по умолчанию кроме как очисткой хранилища.

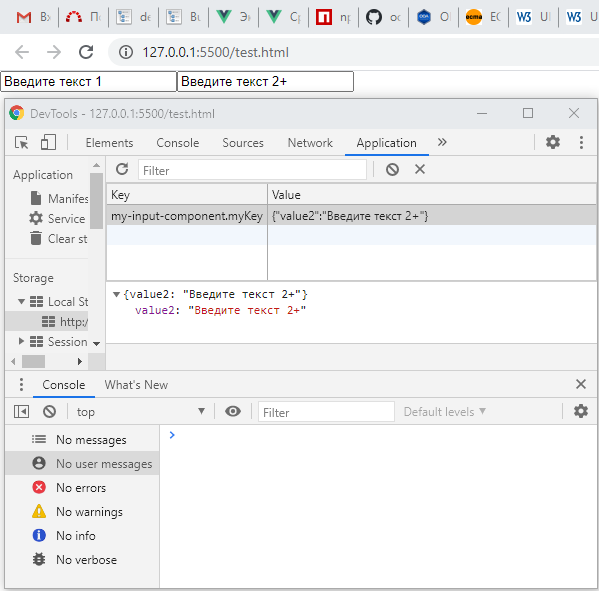
Однако если значение по умолчанию во втором поле изменено и это значение сохранилось в хранилище, то ввод в первое поле значения, совпадающего со значением по умолчанию, работает корректно, а именно, значение первого свойства удаляется из хранилища, тем самым после перезагрузки страницы первое свойство устанавливается в значение по умолчанию.

Например, добавим в оба поля ввода в конец строки символ «**+**»:



Видим, что сохранились оба свойства.

Теперь в первом поле ввода удалим из конца строки символ «**+**», приведя его таким образом к значению по умолчанию:



Видим, что первое свойство удалено из хранилища, и после перезагрузки страницы будет иметь значение по умолчанию.

Теперь во втором поле ввода удалим из конца строки символ «**+**», приведя его таким образом к значению по умолчанию. Однако в этот раз хранилище не обновилось, и ошибка проявилась снова:

