Customizing Increment Arrows on Input of Type Number Using CSS

Asked 2 years, 2 months ago Active 1 year, 2 months ago Viewed 34k times

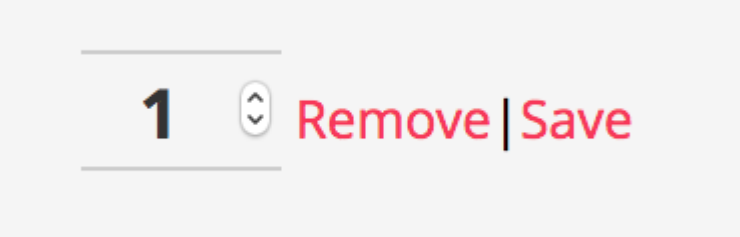
23

19

I have an input of type number that is rendered using the following code:

```
<input class="quantity" id="id_form-0-quantity" min="0" name="form-0-quantity" value="1" type="number">
```

It looks like this:



I would like to turn it into something like this:

The second view is emulated using two separate buttons.

How could I style the arrows as described?

edited Jul 30 '18 at 21:25
Andrei Gheorghiu
41.8k 9 53 82

asked Jul 30 '17 at 4:11
MadPhysicist
1,642 5 14 53

1 Answer

58

The native `input[type=number]` controls are not style-able cross-browser. The easiest and safest way to achieve what you want cross-browser/cross-device is to hide them using:

```
input[type="number"] {
  -webkit-appearance: textfield;
     -moz-appearance: textfield;
          appearance: textfield;
}
input[type=number]::-webkit-inner-spin-button,
input[type=number]::-webkit-outer-spin-button {
  -webkit-appearance: none;
}
```

...which allows you to use your custom buttons, which could be linked to execute the functions the spinners (arrows) would (`.stepUp()` and `.stepDown()`), provided you keep the input's `type="number"` .

For example:

Show code snippet

Note: In order to change the input's value, one needs to find it. To provide flexibility, in the example above I grouped buttons and the `<input>` under a common parent and used that parent to find the `<input>` (choosing not to rely on their proximity or particular order in DOM). The above method ***will change any*** `input[type=number]` ***sibling to the buttons***. If that's not convenient, one could use any other methods to find the input from the buttons:

- by ***id***: `.querySelector('#some-id')` :

```
<button onclick="this.parentNode.querySelector('#some-id').stepUp()"></button>
```

```
<button onclick="this.parentNode.querySelector('.some-class').stepUp()"></button>
```

Also note the above examples only search inside the `.parentNode`, not in the entire `document`, which is also possible:
i.e: `onclick="document.getElementById('#some-id').stepUp()"`

- by ***proximity*** (`previousElementSibling` | `nextElementSibling`)

```
<button onclick="this.previousElementSibling.stepUp()"></button>
```

- any other way to determine and find a particular input element in a DOM structure. For example, one could use third party libraries, such as jQuery:

```
<button onclick="$(this).prev()[0].stepUp()"></button>
```

An important note when using jQuery is that the `stepUp()` and `stepDown()` methods are placed on the DOM element, not on the jQuery wrapper. The DOM element is found inside the `0` property of the `jQuery` wrapper.

Note on `preventDefault()`. Clicking a `<button>` inside a `<form>` *will* trigger the form submission. Therefore, if used as above, inside forms, the `onclick` should also contain `preventDefault();`. Example:

```
<button onclick="$(this).prev()[0].stepUp();preventDefault()"></button>
```

However, if one would use `<a>` tags instead of `<button>`s, this is not necessary. Also, the prevention can be set globally for all form buttons with a small JavaScript snippet:

```
var buttons = document.querySelectorAll('form button:not([type="submit"])');
for (i = 0; i < buttons.length; i++) {
  buttons[i].addEventListener('click', function(e) {
    e.preventDefault();
  });
}
```

... or, using jQuery:

```
}
```

edited Sep 26 '17 at 18:27

answered Jul 30 '17 at 4:26

Andrei Gheorghiu
41.8k 9 53 82

1 @AndreiGheorghiu The CSS to hide the spinners does not seem to work for me in your snippet. I am on Windows 8.1 with Firefox(v54.0.1). – Ricky Dam Jul 30 '17 at 5:24

1 I can't test that now. I'm on Chrome, on Linux. Could you try `input[type=number] { -moz-appearance:textfield; }` and let me know if it works, to add it to answer? It's strange, this worked in all browsers like two months ago or so. I'll have to update it in quite a number of projects... – Andrei Gheorghiu Jul 30 '17 at 5:26

1 @AndreiGheorghiu Your edit for `-moz-appearance` fixed it. Good stuff. And I tried taking it away and the spinners showed up again. So we definitely need the Mozilla vendor prefix to hide it. – Ricky Dam Jul 30 '17 at 5:38

1 @RickyDam, from my point of view `appearance:textfield` makes more sense in this case than `appearance: none;` . I checked on both Chrome and FF and it looks good. The note on MDN's page warns about `appearance` being `non-standard` and that even "none" property doesnt work in all browsers the same. Are they pointing to themselves there? Funny. – Andrei Gheorghiu Jul 30 '17 at 5:45

1 Definitely an odd issue. `-moz-appearance: none;` does not hide the spinners. `-moz-appearance: textfield;` does hide the spinners. CSS is playing mindgames with us ;) – Ricky Dam Jul 30 '17 at 5:51