→ Browser: Document, Events, Interfaces → Forms, controls

6th July 2019

Events: change, input, cut, copy, paste

Let's cover various events that accompany data updates.

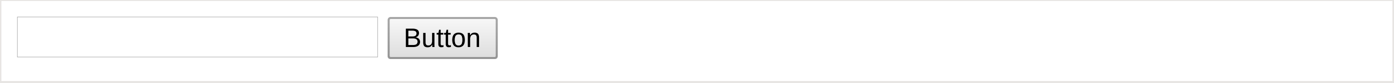
Event: change

The `change` event triggers when the element has finished changing.

For text inputs that means that the event occurs when it loses focus.

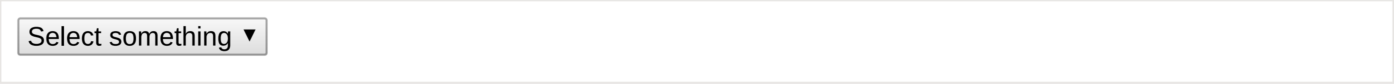
For instance, while we are typing in the text field below – there's no event. But when we move the focus somewhere else, for instance, click on a button – there will be a `change` event:

```
<input type="text" onchange="alert(this.value)">
<input type="button" value="Button">
```



For other elements: `select`, `input type=checkbox/radio` it triggers right after the selection changes:

```
<select onchange="alert(this.value)">
  <option value="">Select something</option>
  <option value="1">Option 1</option>
  <option value="2">Option 2</option>
  <option value="3">Option 3</option>
</select>
```



Event: input

The `input` event triggers every time after a value is modified by the user.

Unlike keyboard events, it triggers on any value change, even those that does not involve keyboard actions: pasting with a mouse or using speech recognition to dictate the text.

For instance:

```
<input type="text" id="input"> oninput: <span id="result"></span>
<script>
  input.oninput = function() {
    result.innerHTML = input.value;
  };
</script>
```

oninput:

If we want to handle every modification of an `<input>` then this event is the best choice.

On the other hand, `input` event doesn't trigger on keyboard input and other actions that do not involve value change, e.g. pressing arrow keys ⇦ ⇨ while in the input.

Can't prevent anything in `oninput`

The `input` event occurs after the value is modified.

So we can't use `event.preventDefault()` there – it's just too late, there would be no effect.

Events: cut, copy, paste

These events occur on cutting/copying/pasting a value.

They belong to ClipboardEvent class and provide access to the data that is copied/pasted.

We also can use `event.preventDefault()` to abort the action, then nothing gets copied/pasted.

For instance, the code below prevents all such events and shows what we are trying to cut/copy/paste:

```
<input type="text" id="input">
<script>
  input.oncut = input.oncopy = input.onpaste = function(event) {
    alert(event.type + ' - ' + event.clipboardData.getData('text/plain'));
    return false;
  };
</script>
```

Please note, that it's possible to copy/paste not just text, but everything. For instance, we can copy a file in the OS file manager, and paste it.

There's a list of methods in the specification that can work with different data types including files, read/write to the clipboard.

But please note that clipboard is a "global" OS-level thing. Most browsers allow read/write access to the clipboard only in the scope of certain user actions for the safety, e.g. in `onclick` event handlers.

Also it's forbidden to generate "custom" clipboard events with `dispatchEvent` in all browsers except Firefox.

Summary

Data change events:

Event	Description	Specials
`change`	A value was changed.	For text inputs triggers on focus loss.
`input`	For text inputs on every change.	Triggers immediately unlike `change` .
`cut/copy/paste`	Cut/copy/paste actions.	The action can be prevented. The `event.clipboardData` property gives read/write access to the clipboard.

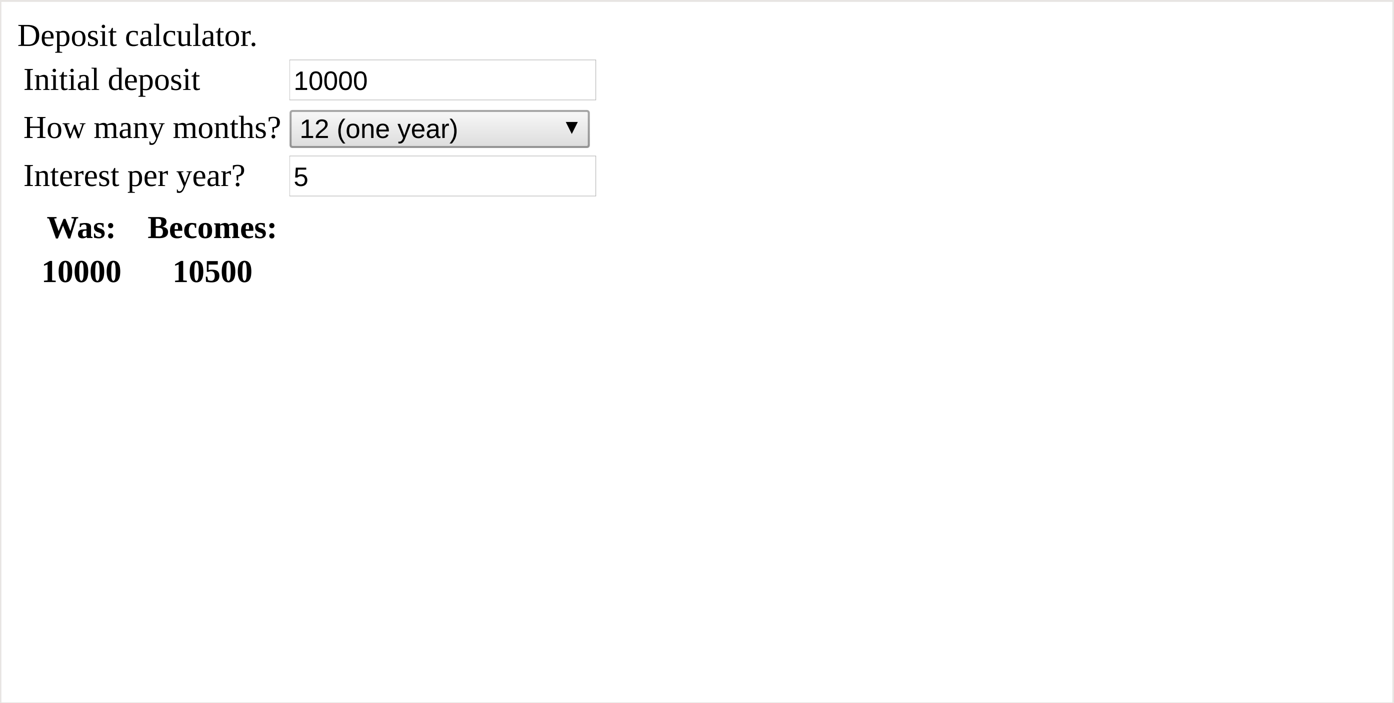
Tasks

Deposit calculator

importance: 5

Create an interface that allows to enter a sum of bank deposit and percentage, then calculates how much it will be after given periods of time.

Here's the demo:



Deposit calculator.

Initial deposit	10000
How many months?	12 (one year) ▾
Interest per year?	5

Was:	Becomes:
10000	10500

Any input change should be processed immediately.

The formula is:

```
// initial: the initial money sum
// interest: e.g. 0.05 means 5% per year
// years: how many years to wait
let result = Math.round(initial * (1 + interest * years));
```

Open a sandbox for the task.

solution

Previous lesson Next lesson

Share

Tutorial map

Comments

- If you have suggestions what to improve - please submit a GitHub issue or a pull request instead of commenting.
- If you can't understand something in the article – please elaborate.
- To insert a few words of code, use the `<code>` tag, for several lines – use `<pre>`, for more than 10 lines – use a sandbox (plnkr, JSBin, codepen…)