the shortcut

Fullstack Web Development Tutorial Lesson 13

Today’s lesson will cover

- Event delegation
- Mous and UI events
- Pop-ups

the shortcut

JavaScript fundamentals

Event Delegation

- Following the concept of Bubbling and Capturing, the idea is instead of having a lot of elements handled, one single handler on ancestor using `event.target` sees where event happened, and handle it
- Could also use classes `.action-save`, `.action-load`, but an attribute `data-action` is better semantically. And we can use it in CSS rules too
- Use event delegation to add “behaviors” to elements *declaratively*, with special attributes and classes. The pattern has two parts:
  - We add a custom attribute to an element that describes its behavior.
  - A document-wide handler tracks events, and if an event happens on an attributed element – performs the action.
- The method `elem.closest(selector)` returns the nearest ancestor that matches the selector.
  - The `closest()` method traverses the `Element` and its parents (heading toward the document root) until it finds a node that matches the provided selector string. Will return itself or the matching ancestor. If no such element exists, it returns `null`.

Mouse events

- The common mouse events are:
    - `mousedown/mouseup`: Mouse button is clicked/released over an element.
    - `mouseover/mouseout`: Mouse pointer comes over/out from an element.
    - `Mousemove`: Every mouse move over an element triggers that event.
    - `click`: Triggers after `mousedown` and then `mouseup` over the same element if the left mouse button was used.
    - `Contextmenu`: Triggers when when the right mouse button is pressed. There are other ways to open a context menu, e.g. using a special keyboard key, it triggers in that case also, so it’s not exactly the mouse event.

Exercise

- Grab the files from github from Task 01 folder
- Create a gallery which allows to change the main featured image by clicking on the thumbnail images below
- By default, click on an anchor element leads to opening up hyperlinks so you will have to prevent this default action to make this work

Events: Data updates - change, input, cut, copy, paste

- The `change` event triggers when the element has finished changing.
- The `input` event triggers every time after a value is modified by the user.
- 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.

Popups and window methods

- **Most browsers block popups if they are called outside of user-triggered event handlers like `onclick`.**
- The syntax to open a popup is: `window.open(url, name, params)`:
    - **Params:** The configuration string for the new window. It contains settings, delimited by a comma. There must be no spaces in params, for instance: `width:200,height=100`.
- Settings for `params`:
    - Position:
        - `left/top` (numeric) – coordinates of the window top-left corner on the screen. There is a limitation: a new window cannot be positioned offscreen.
        - `width/height` (numeric) – width and height of a new window. There is a limit on minimal width/height, so it's impossible to create an invisible window.
    - Window features:
        - `menubar` (yes/no) – shows or hides the browser menu on the new window.
        - `toolbar` (yes/no) – shows or hides the browser navigation bar (back, forward, reload etc) on the new window.
        - `location` (yes/no) – shows or hides the URL field in the new window. FF and IE don't allow to hide it by default.
        - `status` (yes/no) – shows or hides the status bar. Again, most browsers force it to show.
        - `resizable` (yes/no) – allows to disable the resize for the new window. Not recommended.
        - `scrollbars` (yes/no) – allows to disable the scrollbars for the new window. Not recommended.

Exercise

- Grab the HTML doc task02.html from GIthub
- Make the list tree show or hide children nodes
- Requirements:
  - Only one event handler (use delegation): event.target
  - A click outside the node title (on an empty space) should not do anything.
- Wrap every tree node title into `<span>`. Then we can CSS-style them on :hover and handle clicks exactly on text, because `<span>` width is exactly the text width (unlike without it).
- Set a handler to the tree root node and handle clicks on that <span> titles.

Exercise

- Grab the HTML doc task03.html from GIthub
- 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.
- Any **input** and **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));
```

the shortcut

Self Study Assignments

To Dos

- Continue freecodecamp (FCC) Javascript. Ideally finish before we resume after summer.
- Continue with FCC HTML, CSS lessons. Ideally finish all the lessons by end of this month.
- If you believe FCC exercises aren’t the best for you if you are quite advanced already, please start working on your own project and reach out to mentors for help if needed.