

Comp 125 - Visual Information Processing

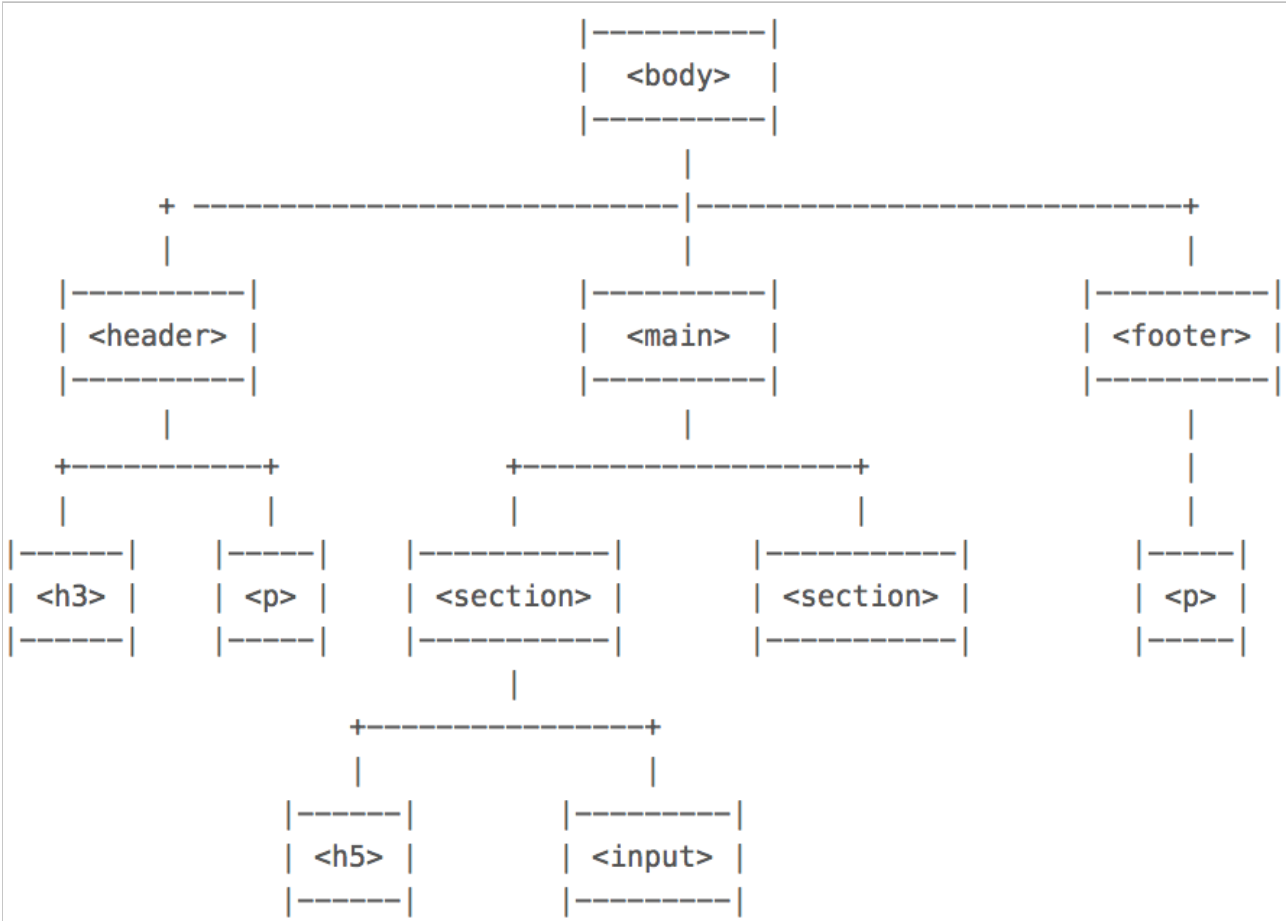
Spring Semester 2019 - Week 14 - Monday

Dr Nick Hayward

HTML5, CSS, & JS - final thoughts

- a basic app that records simple notes
- many additional options we can add
- some basic functionality is needed to make it useful
 - *autosave - otherwise we lose our data each time we refresh the browser*
 - *edit a note*
 - *delete a note*
 - *add author information*
- additional functionality might include
 - *save persistent data to DB, name/value pairs...*
 - *organise and view collections of notes*
 - *add images and other media*
 - *local and APIs*
 - *add contextual information*
 - *again, local and APIs*
 - *structure notes, media, into collection*
 - *define related information*
 - *search, sort...*
 - *export options and sharing...*
- security, testing, design patterns

Image - HTML5, CSS, & JS - DOM recap



Travel Notes - DOM recap

Image - Travel Notes - Series I - recap

travel notes

record notes from various cities and places visited...

add note

add

have fun in St Tropez

ride the tram in Nice

play golf in Mougins

app's copyright information, additional links...

Travel Notes - Series I - Demo 8 recap

HTML5, CSS, & JS - example - add-ons

new features and add-ons...

- delete all notes
- delete a single note
- new event handlers
- additional styling

HTML5, CSS, & JS - example - part 1.1

delete option - all notes

- standard `remove()` function in jQuery

```
$("#p").remove();
```

- option to **delete all** notes from `.note-output`
- add a new toolbar for note controls and options

```
<section class="note-controls">  
  <button id="notes-delete">Delete all</button>  
</section>
```

- then add some simple styling for this new toolbar

```
/* note controls */  
.note-controls {  
  margin: 10px 0 10px 0;  
  padding: 2px;  
  border-bottom: 1px solid #dedede;  
  display: none;  
}  
/* simplify default button styles for note controls */  
.note-controls button {  
  padding: 2px;  
  margin: 2px;  
  border-radius: 0;  
  border: 1px solid #dedede;  
  cursor: pointer;  
}
```

HTML5, CSS, & JS - example - part 1.2

delete option - all notes - plain js

```
// delete all notes button
let deleteAll = document.getElementById('notes-delete');

// add event listener for delete all notes...
deleteAll.addEventListener('click', () => {
  // get notes from DOM
  let notes = noteOutput.querySelectorAll('p');
  // loop through notes and remove a single note per iteration...
  for (let note of notes) {
    note.remove();
  }
});
```

HTML5, CSS, & JS - example - part 2.1

delete option - all notes

- note controls toolbar is hidden, by default in the CSS
- need some way to check its visibility as we add our notes
 - *no notes, then the toolbar is not required*

```
//check element visibility - expects single element relative to display:none  
function checkVisible(element) {  
  if (element.is(":hidden")) {  
    element.fadeIn();  
  }  
}
```

- simply checking a passed element to see whether it is hidden
 - *then fadeIn() as necessary*
- can update this method later on to check hidden and visible
- call this function as required

```
checkVisible($(".note-controls"));
```


HTML5, CSS, & JS - example - part 2.2

delete option - all notes - plain js

- use `display` property to check node

```
// check visibility of passed node
function checkVisible(node) {
  // check passed node's current visibility
  if (node.style.display !== 'block') {
    // show in DOM to allow fadeIn...
    node.style.display = 'block';
    // call fadeIn for node in DOM
    fadeIn(node);
  }
}
```

- & usage with a defined node

```
// define node to check...
let controls = document.getElementById('controls');
// call function
checkVisible(controls);
```

HTML5, CSS, & JS - example - part 2.3

delete option - all notes - plain js

- use `visibility` property to check node

```
// check visibility of passed node  
function checkVisible(node) {  
    // check passed node's current visibility  
    if (node.style.visibility = 'hidden') {  
        // show in DOM to allow fadeIn...  
        node.style.display = 'block';  
        node.style.visibility = 'visible';  
        // call fadeIn for node in DOM  
        fadeIn(node);  
    }  
}
```

HTML5, CSS, & JS - example - part 3

delete option - all notes

- add a note, the `.note-controls` toolbar is shown
 - ***delete all*** button now becomes available to our users

```
//handle deletion of all notes
$("#notes-delete").on("click", function(e) {
  var $note = $(".note-output p");
  $(this).parent().hide();
  $note.remove();
});
```

- creating a new handler for the click events on the `#notes-delete` button
- hides its own container, the notes toolbar
- then removes all of the notes, `p`, from the `.note-output` section

HTML5, CSS, & JS - example - part 4.1

JS code so far

```
//check element visibility - expects single element relative to display:none
function checkVisible(element) {
  if (element.is(":hidden")) {
    element.fadeIn();
  }
}
...
//handle deletion of all notes
$("#notes-delete").on("click", function(e) {
  var $note = $(".note-output p");
  $(this).parent().hide();
  $note.remove();
});
```

HTML5, CSS, & JS - example - part 4.2

JS code so far - plain JS

- hide parent node for controls...

```
// delete all notes button
let deleteAll = document.getElementById('notes-delete');

// add event listener for delete all notes...
deleteAll.addEventListener('click', () => {
  // hide parent controls node...
  deleteAll.parentNode.style.display = 'none';
  // get notes from DOM
  let notes = noteOutput.querySelectorAll('p');
  // loop through notes and remove a single note per iteration...
  for (let note of notes) {
    // remove single node
    note.remove();
  }
});
```

- DEMO 1 - travel notes - series 2

HTML5, CSS, & JS - example - part 5

delete option - all notes

- still making an assumption notes exist in the note-output section
- add an additional function to check element exists in the DOM or not
- use `length` property - plain JS & jQuery

```
$("p").length
```

- new function for checking elements in the DOM - plain JS & jQuery

```
//check elements exists
function checkExist(element) {
  if (element.length) {
    return true;
  } else {
    return false;
  }
}
```

HTML5, CSS, & JS - example - part 6.1

delete option - all notes

- updated delete all notes option to include check for notes
- call `checkExist()` function in conditional statement

```
//handle deletion of all notes
$("#notes-delete").on("click", function(e) {
  //set note selector
  var $note = $(".note-output p");
  //check $note exists
  if (checkExist($note) === true) {
    //hide note-controls
    $(this).parent().hide();
    //remove all notes
    $note.remove();
  }
});
```

HTML5, CSS, & JS - example - part 6.2

delete option - all notes - plain JS

```
// add event listener for delete all notes...
deleteAll.addEventListener('click', () => {
  // get notes from DOM
  let notes = noteOutput.querySelectorAll('p');
  // check notes in DOM
  if (checkExist(notes) === true) {
    // hide parent controls node...
    deleteAll.parentNode.style.display = 'none';
    // loop through notes and remove a single note per iteration...
    for (let note of notes) {
      // remove single node
      note.remove();
    }
  }
});
```

- DEMO 2 - travel notes - series 2

Image - Travel Notes - Series 2 - demo 2

travel notes

record notes from various cities and places visited...

add note

add

Delete all

stroll along the Promenade des Anglais in Nice

lose money in Monaco

meet Picasso in Antibes

be seen in Cannes

app's copyright information, additional links...

Travel Notes - Series 2 - Demo 2

HTML5, CSS, & JS - example - part 7

delete option - per note

- consider adding a single delete option per note
- allowing a user to selectively delete their chosen note
 - *regardless of hierarchical position within the .note-output section*
- design decisions for such an option might include
 - *do we offer a selection option, such as checkboxes, to select one or more delete items*
 - *perhaps a single delete button per note*
 - *a drag and drop to delete option*
 - *there are many different ways to present and use this option*
- programmatically follow a similar pattern for deletion of the note
- three jQuery functions can help us remove elements from a document
 - *remove()*
 - *detach()*
 - *replaceWith()*

jQuery - removing elements - quick overview

- used `remove()` function with delete all notes
 - *best used to remove elements permanently from a document*
 - *will **unbind** any attached event handlers for elements being removed*
 - *will return reference to removed elements, but not the original bound events*
- `detach()` often used for any temporary removal requirements
 - *eg: update a lot of the DOM, detach affected elements, then insert later...*
 - *retains its event handlers, and we can add these elements later*

```
$("#p").detach();
```

- then append the attached elements as required

```
var $detachP = $("#p").detach();  
$detachP.appendTo("#detached");
```

- `replaceWith()` replaces an element, or group of elements, with passed element
- event handlers for the replaced elements are unbound

```
var $replacedP = $(".note-output p").first().replaceWith("<p>replaced...</p>");
```

HTML5, CSS, & JS - example - part 8.1

delete option - per note

- simply need to delete the selected note
 - *use the same `remove()` function for single and all notes*
- add option per note to allow user to delete a required note
- add a delete button for each note
 - *add programmatically with each new note*

```
function createButton(buttonClass, buttonText) {  
  var $button = $('<button class="'+buttonClass+'">'+buttonText+'</button>');  
  return $button;  
}
```

- new function allows us to create simple buttons as required
 - *a specified class and button text passed as parameters*
 - *use function to build required delete button in `createNote()` function*

```
//create delete button  
var $delete_button = createButton("note-delete", "delete");
```

HTML5, CSS, & JS - example - part 8.2

delete option - per note - plain js

```
// create button element - pass class and text
function createButton(btnClass, btnTxt) {
  // create button node
  let btnNode = document.createElement('button');
  // create button text node
  let btnTxtNode = document.createTextNode(btnTxt);
  // set attribute on button node
  btnNode.setAttribute('class', btnClass);
  // append text to button
  btnNode.appendChild(btnTxtNode);
  // return new button node with text and attribute...
  return btnNode;
}
```

- then call as required,

```
// create delete button for note
let delButton = createButton('note-delete', 'delete');
```

HTML5, CSS, & JS - example - part 9.1

delete option - per note

- append delete option to note
 - *before adding note to the DOM in createNote function*

```
function createNote() {  
    ...  
    //set content for note  
    $note.html($note_text.val());  
    //append delete button to each note  
    $note.append($delete_button);  
    ...  
}
```

HTML5, CSS, & JS - example - part 9.2

delete option - per note - plain js

```
function createNote(input, output) {
  // get value from input field for note
  let inputVal = input.value;

  // check input value
  if (inputVal !== '') {
    // create p node
    let p = document.createElement('p');
    // create delete button for note
    let delButton = createButton('note-delete', 'delete');
    // prepend button to note
    p.prepend(delButton);
    // create text node
    let noteText = document.createTextNode(inputVal);
    // append text to paragraph
    p.appendChild(noteText);
    // append new paragraph and text to existing note output
    output.appendChild(p);
    // call custom animation for fade in...
    //fadeIn(p);
    // clear input text field
    input.value = '';
  }

  let controls = document.getElementById('app-controls');
  checkVisible(controls);
}
```

HTML5, CSS, & JS - example - part 10

delete option - per note

- with jQuery
 - need to bind a click event to the dynamically created delete note button
 - plain JS option simpler
- delete button is being added to the DOM dynamically
 - need to add handler for single note deletion event to existing DOM element
 - add handler to parent `.note-output` and then new `button.note-delete`

```
$(".note-output").on("click", "button.note-delete" , function() {  
    //delete parent note  
    $(this).parent().remove();  
    //set note selector  
    var $note = $(".note-output p");  
    //check for empty notes, and then remove note-controls  
    if (checkExist($note) === false) {  
        //hide note-controls  
        $(".note-controls").hide();  
    }  
});
```

- DEMO 3 - travel notes - series 2 - jQuery

Image - Travel Notes - Series 2 - demo 3

travel notes

record notes from various cities and places visited...

add note

breakfast in Antibes

lunch in Nice

dinner in Monaco

app's copyright information, additional links...

Travel Notes - Series 2 - Demo 3

HTML5, CSS, & JS - example - part II

delete option - per note

- now allow our users to delete a single note
- single note option is awkward at the moment
- simply allow a user to either mouseover or select a note to show additional options
 - *showing the available delete button*
- enable a user to select their note of choice
 - *need to bind a click event to a note*

```
//handle click event per note
$(".note-output").on("click", "p", function() {
  ...
});
```

- user selects a note
 - *no check for previous other visible delete buttons*
 - *ensure only delete button for selected note is shown*

Image - HTML5, CSS, & JS - too many delete buttons

travel notes

record notes from various cities and places visited...

add note

cannes note

nice note

monaco note

antibes note

app's copyright information, additional links...

Travel Notes - Week 6 - Too many delete buttons