Comp 125 - Visual Information Processing

Spring Semester 2019 - Week 4 - Wednesday

Dr Nick Hayward

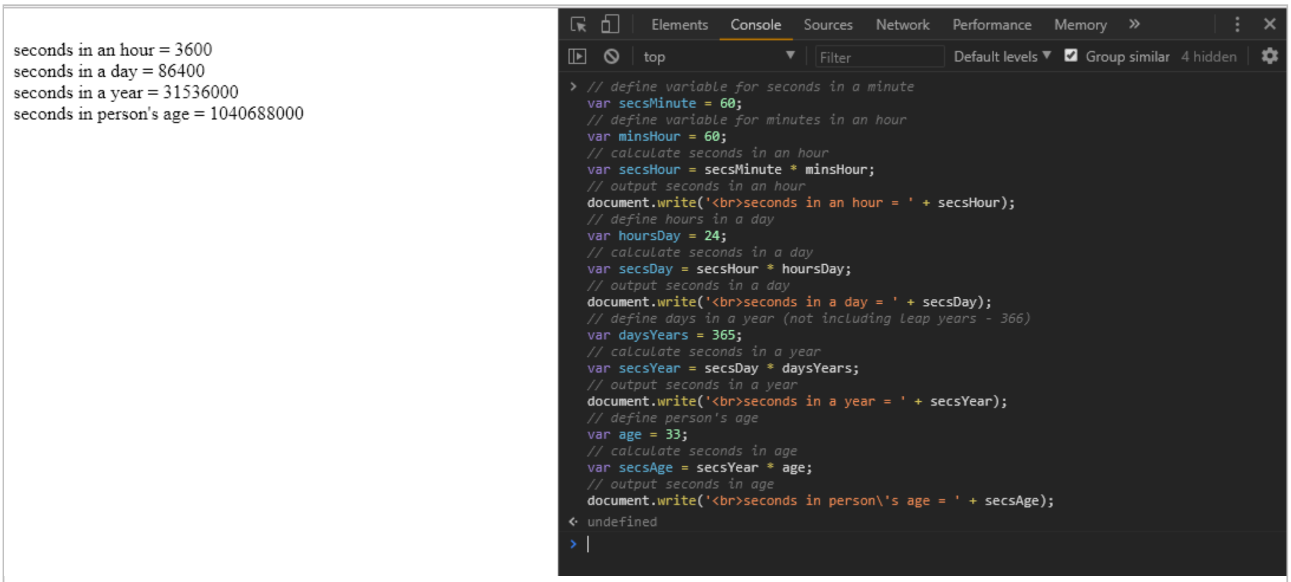
Fun exercise - using variables and operators

- calculate the **number of seconds in an hour**
- using the **number of seconds in an hour**, calculate the **number of seconds in a day**
- using **number of seconds in a day**, calculate the **number of seconds in a year**
- using **number of seconds in a year**, calculate the **number of seconds in your current age** in years, e.g. 22 years

Output each answer to the document with a line break between each result.

Fun exercise - using variables and operators

one possible answer using current notes...



```
seconds in an hour = 3600
seconds in a day = 86400
seconds in a year = 31536000
seconds in person's age = 1040688000
```

```
> // define variable for seconds in a minute
  var secsMinute = 60;
  // define variable for minutes in an hour
  var minsHour = 60;
  // calculate seconds in an hour
  var secsHour = secsMinute * minsHour;
  // output seconds in an hour
  document.write('<br>seconds in an hour = ' + secsHour);
  // define hours in a day
  var hoursDay = 24;
  // calculate seconds in a day
  var secsDay = secsHour * hoursDay;
  // output seconds in a day
  document.write('<br>seconds in a day = ' + secsDay);
  // define days in a year (not including leap years - 366)
  var daysYears = 365;
  // calculate seconds in a year
  var secsYear = secsDay * daysYears;
  // output seconds in a year
  document.write('<br>seconds in a year = ' + secsYear);
  // define person's age
  var age = 33;
  // calculate seconds in age
  var secsAge = secsYear * age;
  // output seconds in age
  document.write('<br>seconds in person\'s age = ' + secsAge);
<· undefined
> |
```

Fun Exercise - using variables and operators

HTML - Intro

- acronym for *HyperText Markup Language*
- simple way to structure visual components of a website or web application
- HTML also uses keywords, or element tags
  - *follow a defined syntax*
- helps us to create web pages and web applications
  - *web browsers, such as Chrome or Firefox, may render for viewing*
- an error can stop a web page from rendering
  - *more likely it will simply cause incorrect page rendering*
- interested in understanding the core of web page designing
  - *understand at least the basics of using HTML*

HTML - structure of HTML

- basic HTML tag defines the entire HTML document

```
<html>
  ...
</html>
```

```
<!DOCTYPE html>
<html>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

HTML - Element syntax - part 1

Constructed using elements and attributes, which are embedded within an HTML document.

Elements should adhere to the following,

- start with an opening element tag, and close with a matching closing tag
  - *names may use characters in the range **0-9**, **a-z**, **A-Z***
- content is, effectively, everything between opening and closing element tags
- elements may contain empty or *void* content
- empty elements should be closed in the opening tag
- most elements permit attributes within the opening tag

HTML - Element syntax - part 2

An element's *start* tag adheres to a structured pattern, which may be as follows,

1. a **<** character
2. tag name
3. optional **attributes**, which are separated by a space character
4. optional space characters (one or more...)
5. optional **/** character, indicating a **void** element
6. a **>** character

For example,

```
<!-- opening element tag -->
<div>
<!-- void element - XHTML -->
<br />
<!-- void element - HTML5 -->
<br>
```

HTML - Element syntax - part 3

An element's *end* tag also adheres to a pattern, again exactly as defined as following,

1. a **<** character
2. a **/** character
3. element's tag name (i.e. name used in matching start tag)
4. optional space characters (one or more...)
5. a **>** character

For example,

```
<!-- element's matching end tag -->
</div>
```

NB: void elements, such as `<br>` or `<img  />`, do *not* specify end tags.

HTML - Element syntax - part 4

- HTML, XHTML, can be written to follow the patterns and layouts of XML
- HTML elements can also be nested with a parent, child, sibling...
    - *relationship within the overall tree data structure for the document*
- as the HTML page is loaded by a web browser
    - *the HTML DOM (document object model) is created*
- basically a tree of objects that constitutes the underlying structure
    - *the rendered HTML page*
- DOM gives us an API (application programming interface)
    - *a known way of accessing, manipulating the underlying elements, attributes, and content*
- DOM very useful for JavaScript manipulation

HTML - attribute syntax - part 1

- HTML attributes follow the same design pattern as XML
- provide additional information to the parent element
- placed in the opening tag of the element
- follow the standard syntax of name and value pairs
- many different permitted legal attributes in HTML
- four common names that are permitted within most HTML elements
  - *`class`*, *`id`*, *`style`*, *`title`*

HTML - attribute syntax - part 2

Four common names permitted within most HTML elements

- `class`
  - *specifies a classname for an element*
- `id`
  - *specifies a unique ID for an element*
- `style`
  - *specifies an inline style for an element*
- `title`
  - *specifies extra information about an element*
  - *can be displayed as a tooltip by default*

NB:

- cannot use same name for two or more attributes
  - *regardless of case*
  - *on the same element start tag*

HTML - attribute syntax - part 3

A few naming rules for attributes

- empty attribute syntax
  - *`<input disable>`*
- unquoted attribute-value syntax
  - *`<input value=yes>`*
  - *value followed by /, at least one space character after the value and before /*
  - *i.e. usage with a void element...*
- single quoted attribute-value syntax
  - *`<input type='checkbox'>`*
- double quoted attribute-value syntax
  - *`<input title="hello">`*

n.b.

- further specific restrictions may apply for the above
- consult W3 Docs for further details
- above examples taken from W3 Docs - Syntax Attributes Single Quoted

HTML - Doctype - HTML5

- `DOCTYPE` is a special instruction to the web browser
  - *concerning the required processing mode for rendering the document's HTML*
- doctype is a required part of the HTML document
- first part of our HTML document
- should always be included at the top of a HTML document, e.g.

```
<!DOCTYPE html>
```

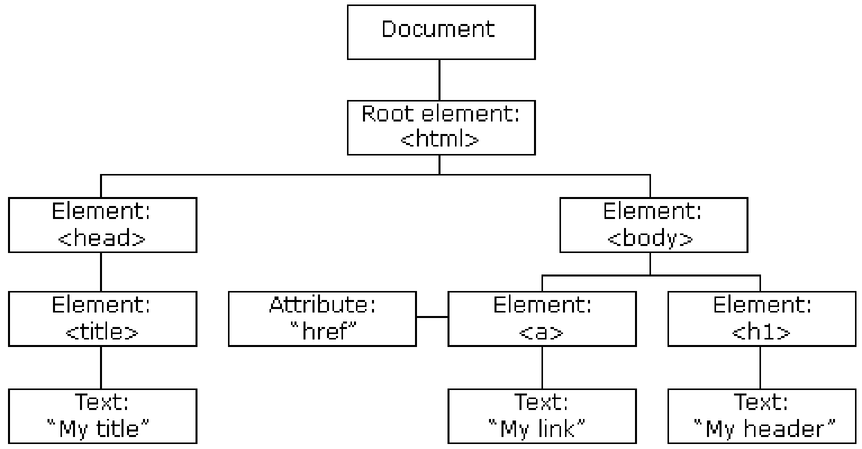
or

```
<!doctype html>
```

- doctype we add for HTML5 rendering
- not a HTML element, simply tells the browser required HTML version for rendering

DOM Basics - intro

A brief introduction to the document object model (DOM)



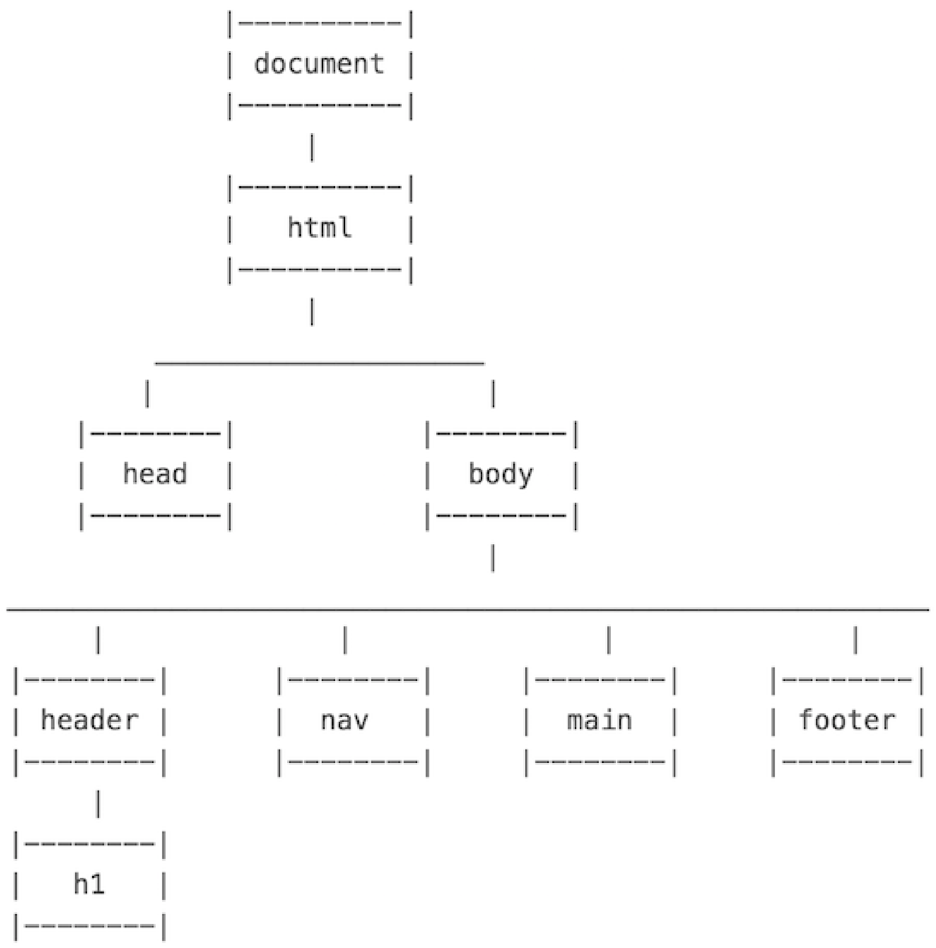
HTML DOM

- Source - W3Schools - JS HTML DOM

DOM Basics - what is DOM?

- **DOM** is a platform and language independent way
  - *to access and manipulate underlying structure of HTML document*
- structured as a representation of a tree data structure
  - *its manipulation follows this same, standard principle*
- DOM tree is constructed using a set of nodes
  - *tree is designed as a hierarchical representation of the underlying document*
- each node on our tree is an element within our HTML document
- inherent hierarchical order originates with the **root** element
  - ***root** sits at the top of our **tree***
  - *descends down following lineage from node to node*
- each node is a child to its parent
  - *we can find many siblings per node as well*
- root at the top of the tree...

Image - HTML DOM



```
             |----------|
             | document |
             |----------|
                  |
             |----------|
             |   html   |
             |----------|
                  |
         ____________________
        |                    |
    |--------|           |--------|
    |  head  |           |  body  |
    |--------|           |--------|
                             |
_________________________________________________________
     |              |               |              |
|--------|      |--------|     |--------|     |--------|
| header |      |  nav   |     |  main  |     | footer |
|--------|      |--------|     |--------|     |--------|
     |
|--------|
|   h1   |
|--------|
```

HTML DOM

DOM Basics - useful elements

element tag	usage & description
<html>	container element for a HTML document
<head>	contains metadata and document information
<body>	contains main content rendered as the HTML document
<header>	page header...
<nav>	navigation, stores and defines a set of links for internal or external navigation
<main>	defined primary content area of document
<footer>	page footer...
<section>	a section of a page or document
<article>	suitable for organising and containing independent content
<aside>	defines content aside from the content which contains this element
<figure>	logical grouping of image and caption
<img>	image - can be local or remote using url in src attribute
<figcaption>	image caption
<h1>, <h2>...	headings from 1 to 6 (1 = largest)
<a>	anchor - link to another anchor, document, site...
<p>	paragraph
<ul>, <ol>, <dl>	unordered, ordered, definition lists
<li>	list item, used with <ul>, <ol>...
<dt>	definition term, used with <dl>
<dd>	definition description, used with <dl>
<table>	standard table with rows, columns...
<tr> >	table row, used with <table>
<th>	table heading, used with <table> and child to <tr>
<td>	table cell, used with <table> and child to <tr>

element tag	usage & description
`<div>`	non-semantic container for content, similar concept to `<section>`
`<span>`	group inline elements in a HTML document
`<canvas>`	HTML5 element for drawing on the HTML page
`<video>`	HTML5 element for embedding video playback
`<audio>`	HTML5 element for embedding audio playback

NB: *`<div>` and `<span>` can be used as identifiers when there is no other suitable element to define parts of a HTML5 document. e.g. if there is no defined or significant semantic meaning...*