

## **Comp 125 - Visual Information Processing**

---

Spring Semester 2018 - week 4 - wednesday

Dr Nick Hayward

## JS Objects - combine arrays and objects

---

- objects and arrays may also be combined in JavaScript
  - *an object in an array, array in object...*

```
// create array with object
var archives = [
  { name: 'waldzell', access: 'castalia', purpose: 'gaming' },
  { name: 'bodleian', access: 'oxford', purpose: 'research' }
];
```

- then access inner object

```
// get first archive object
var firstArchive = archives[0];
```

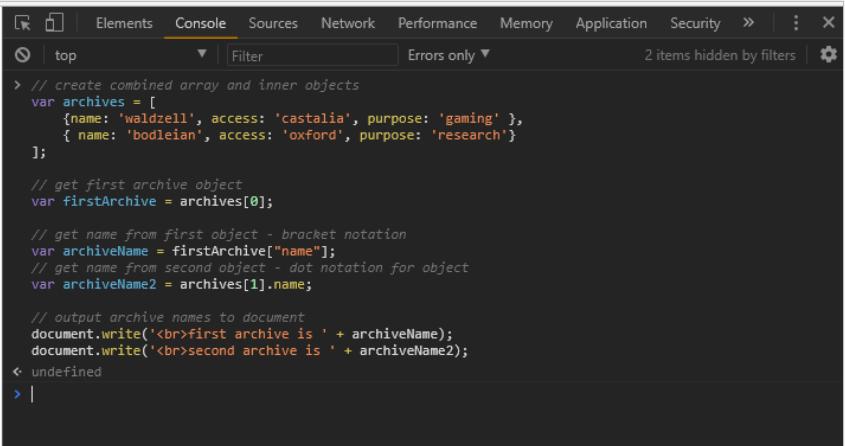
- then, we can get the name of the first archive, e.g.

```
// get name from first object - bracket notation
var archiveName = firstArchive["name"];
// get name from second object - dot notation for object
var archiveName2 = archives[1].name;
```

## JS Objects - combine arrays and objects

combine arrays and objects...access inner values

first archive is waldzell  
second archive is bodleian



```
> // create combined array and inner objects
var archives = [
  {name: 'waldzell', access: 'castalia', purpose: 'gaming' },
  { name: 'bodleian', access: 'oxford', purpose: 'research'}
];

// get first archive object
var firstArchive = archives[0];

// get name from first object - bracket notation
var archiveName = firstArchive["name"];
// get name from second object - dot notation for object
var archiveName2 = archives[1].name;

// output archive names to document
document.write('<br>first archive is ' + archiveName);
document.write('<br>second archive is ' + archiveName2);
< undefined
> |
```

JS - array and object combined

## Fun exercise - using objects

---

- create an object or objects with information about an archive
  - *include name and location of the archive*
- use a combination of arrays and objects to store information about books in the archive - minimum five books
  - *include author's name, book title, date of publication, number of pages...*
- output to the document all of the names of the books in the archive
  - *output to the document all information for at least one book in the archive*

Output answers to the document with link breaks between results.

# 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 I

---

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 I

---

- 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

## References

- [W3 Docs - Syntax](#)
- [W3 Docs - Attribute Syntax](#)
- [W3 Docs - Syntax Attributes Single Quoted](#)