CRITICAL LOG

Web Authoring Item 2

Jack Steel
STE15602932

Contents

Report

During the initial design phase, I considered how to structure the website regarding the file organisation. Since this site is relatively small (less than ten separate pages) I decided to 'root' all the HTML files; this simplifies things when referencing files because all the HTML files are stored in the root directory. Any file paths used within the HTML file will be much simpler, in addition to any in the JavaScript files since file paths in JavaScript are relative from the HTML page to which they are linked.

For a larger site, it would be better to use a more hierarchical structure for the HTML files, having many 'index' files in many directory levels. I have used most of the best practice and standards from the workshops, e.g. lowercase HTML tags, using HTML5 elements like 'article' and 'aside' instead of simple divs. For JavaScript, to be consistent, I had to decide which naming conventions to use, and research and apply where possible the W3 Schools' best practice guidelines (w3schools.com, 2016). For naming conventions, I decided on Camel case for function and variable identifiers, all caps for constants, and Pascal case for class identifiers, as these are what I am most used to using in other languages.

Web standards have changed dramatically since the foundation of World Wide Web Consortium (W3C). In the late 1990s and early 2000s, Microsoft's Internet Explorer dominated the web browser market (The history of the web - W3C Wiki, 2014). At this time the specifications of the W3C were still just considered recommendations and browsers rarely conformed to any significant amount of them (ibid.). This lack of consensus across browsers gave companies that made browsers, especially Microsoft (due to IE's dominance) the ability to significantly tailor their websites for their browsers so that a Microsoft site might load much faster or have more useful features on Internet Explorer than any other browser. This bias is more of a weakness with the lack of popular browsers' support for the specification than the specification itself. Currently, it is expected that browsers conform to the W3C's standards as much as possible, meaning developers do not have to worry as much about making their site compatible with different browsers. Sticking to the web standards ensures a mostly uniform outcome across browsers, at least for basic functionality. The main advantage of current web standards over older ones is they are all backwards compatible. This means a browser that supports HTML5 will also inherently support HTML 4.01. Whereas the W3C specification for XHTML 2.0 was not backwards compatible with existing markup, this ultimately lead to XHTML 2.0 being a failure and never being adopted as a standard.

To maximise my site's interoperability, I have used several techniques, the most obvious of which is a responsive design; wherever practical I have used relative values for sizes and distances. I have also implemented CSS media queries, particularly for the navigation bar, changing it to a dropdown menu when the screen is too small. This ensures that the site is readily usable and looks good on any standard resolution screen. Less evident is the use of several prefixes for certain CSS properties; different prefixes are used by various browsers to perform essentially the same thing, including them all, ensures that it works on all of them. I have also had to do a similar thing in JavaScript at one point, using two lines of code that do the same thing in different browsers because Firefox implements it slightly differently to others. To go further with this, it would be best to use separate style sheets for different browsers, especially Internet Explorer (due to its lack of support), to best account for the slightly different implementations of each. However, this falls outside the scope of this project. If this were a real company website, I would also recommend using a server-side scripting language instead of client-side JavaScript. Server-side scripting is more secure than client-side, and doing the logic on the server also has the advantage of being able to much more easily use a database to store details for user accounts, enabling users to access their shopping cart from many different devices.

Another 'best practice' that I elected to ignore in this case is 'minification' and 'compilation' of source code. This process is typically applied only to the live website as it makes development much more challenging. Compilation involves compiling JavaScript from several files into one JavaScript file. This differs from the traditional use of compilation as JavaScript is an interpreted language and cannot be compiled into an executable file such as C#. Compilation is useful as with lots of small files the speed bottleneck is the time it takes to generate and process the HTTP request, not the total download time of the files. Compilation minimises the number of HTTP requests needed (Compressing your JavaScript with closure compiler, 2015). Minification removes any unnecessary characters from files, such as whitespace and comments, and even renames variables to single characters when possible (Bazon, 2012). Doing this reduces the file size and thus reduces download times for relatively large files. While both of these processes can significantly improve a site's performance, they also make it difficult to maintain and update if an original version is not kept.

I chose to use an object-oriented style for my JavaScript. Object-oriented programming helps to create cleaner more understandable code, which can easily be reused for other cases, due to encapsulation. Using classes and objects helped me structure my code in a way that is much easier to maintain.

Website Link

http://jacksteel.azurewebsites.net

demo video: http://jacksteel.azurewebsites.net/demo.html

W3C Markup Validation

HTML

Air.html

Showing results for uploaded file air.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. **Warning** **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**

From line 68, column 5; to line 68, column 33

```
pt-->↵    <section id="productSection">↵
```

Contact.html

Showing results for contact.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. **Warning** **Article lacks heading. Consider using** h2 - h6 **elements to** **add identifying headings to all articles.**

From line 49, column 3; to line 49, column 24

```
</div>↵                <article id="details">↵                <u
```

Events.html

Showing results for events.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. **Warning** **Article lacks heading. Consider using** h2 - h6 **elements to** **add identifying headings to all articles.**

From line 51, column 5; to line 51, column 34

```
i" />↵                <article class="eventContent">↵
```

2. **Warning** **Article lacks heading. Consider using** h2 - h6 **elements to** **add identifying headings to all articles.**

From line 61, column 5; to line 61, column 34

```
n" />↵                <article class="eventContent">↵
```

3. **Warning** **Article lacks heading. Consider using** h2 - h6 **elements to** **add identifying headings to all articles.**

From line 71, column 5; to line 71, column 34

```
s" />↵                <article class="eventContent">↵
```

Index.html

Showing results for index.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. **Warning** **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
   From line 84, column 3; to line 84, column 34
   `ucts-->↵ <section id="featuredSlideShow">↵ <d`

2. **Warning** **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
   From line 74, column 2; to line 74, column 27
   `page-->↵ <section id="mainContent">↵ <!-`

3. **Warning** **Article lacks heading. Consider using h2-h6 elements to add identifying headings to all articles.**
   From line 103, column 3; to line 103, column 11
   `</div>↵ <article>↵ <p`

Land.html

Showing results for land.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. **Warning** **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
   From line 67, column 5; to line 67, column 33
   `pt-->↵ <section id="productSection">↵`

Productdetail.html

Showing results for productdetail.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. Warning **Consider using the h1 element as a top-level heading only (all h1 elements are treated as top-level headings by many screen readers and other tools).**
   From line 68, column 6; to line 68, column 23
   `le">↵ <h1 id="prodName">Produ`
2. Warning **Consider using the h1 element as a top-level heading only (all h1 elements are treated as top-level headings by many screen readers and other tools).**
   From line 75, column 6; to line 75, column 9
   `le">↵ <h1>Total<`
3. Warning **Consider using the h1 element as a top-level heading only (all h1 elements are treated as top-level headings by many screen readers and other tools).**
   From line 83, column 6; to line 83, column 9
   `le">↵ <h1>Custom`
4. Warning **The color input type is not supported in all browsers. Please be sure to test, and consider using a polyfill.**
   From line 91, column 7; to line 91, column 62
   `el>↵ <input id="colourPicker" type="color" value="#ff0000" /> </li>`
5. Warning **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
   From line 64, column 3; to line 64, column 24
   `tion-->↵ <section id="details">↵ <!`
6. Warning **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
   From line 55, column 2; to line 55, column 10
   `ction-->↵ <section>↵ <!-`

Sea.html

Showing results for sea.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. Warning **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
   From line 66, column 5; to line 66, column 33
   `pt-->↵ <section id="productSection">↵`

Shoppingcart.html

Showing results for shoppingcart.html

Checker Input

Show source outline image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

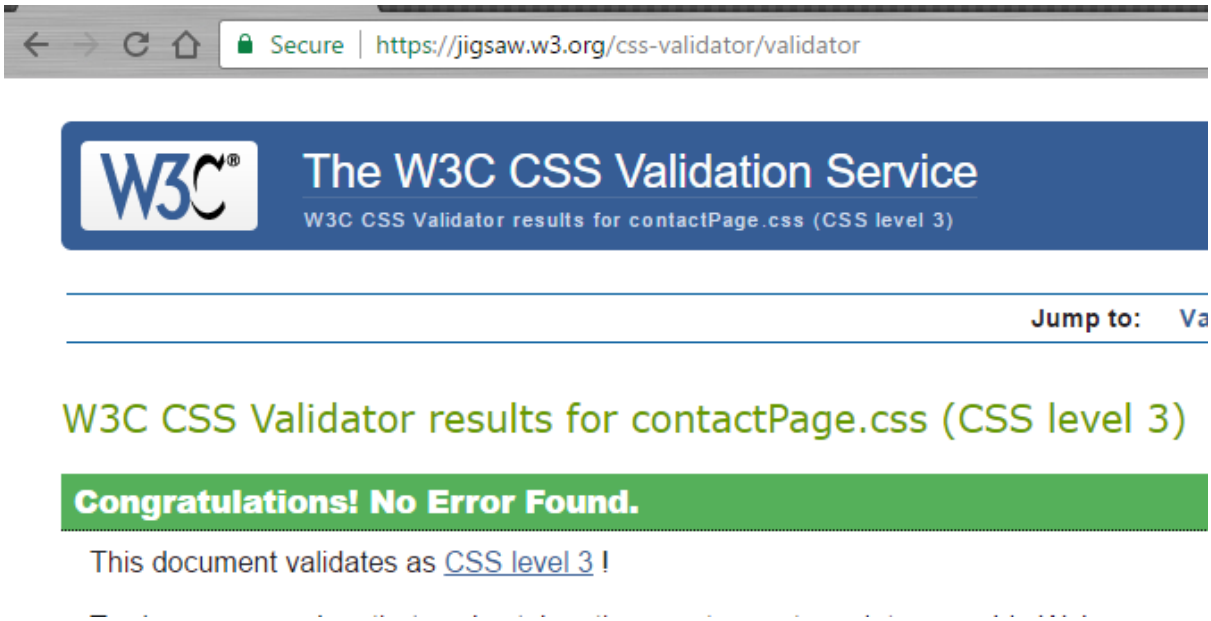
Message Filtering

1. **Warning** **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
From line 50, column 3; to line 50, column 26
`ection>↵ <section id="cartItems">↵ <!`

2. **Warning** **Section lacks heading. Consider using h2-h6 elements to add identifying headings to all sections.**
From line 49, column 2; to line 49, column 10
`/header>↵ <section>↵ <se`

CSS

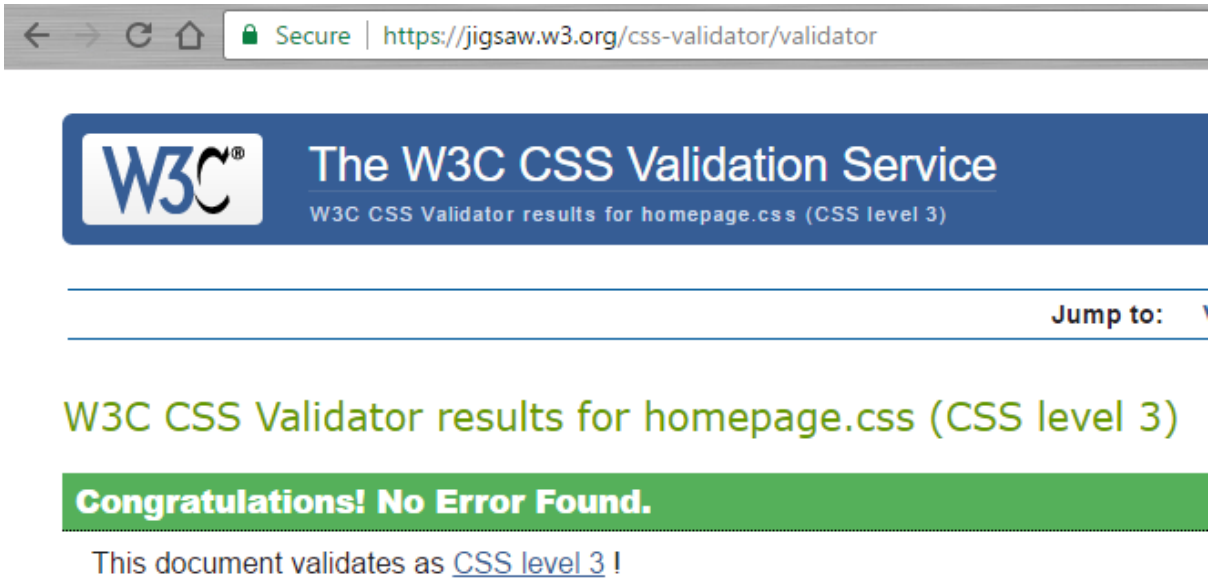
Contactpage.css



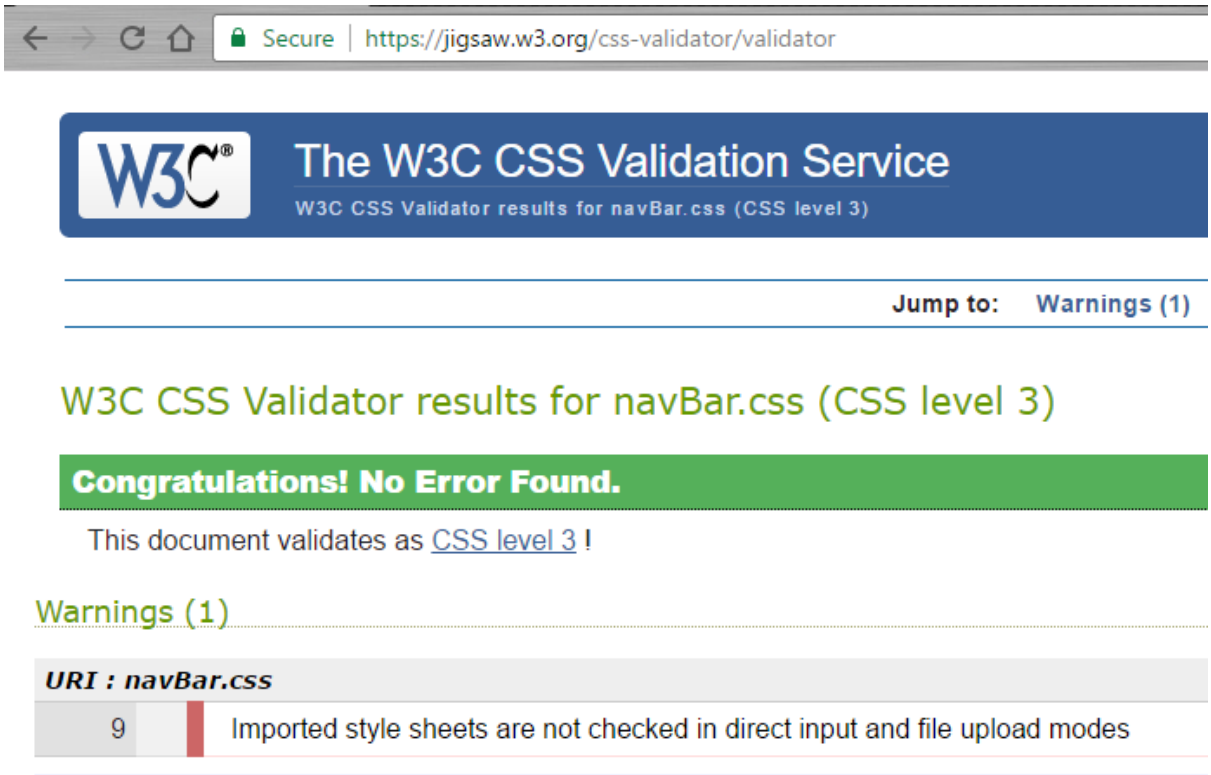
Events.css

Footer.css

Hompage.css



Navbar.css


Secure | https://jigsaw.w3.org/css-validator/validator
The W3C CSS Validation Service
W3C CSS Validator results for navBar.css (CSS level 3)
Jump to: Warnings (1)
W3C CSS Validator results for navBar.css (CSS level 3)
Congratulations! No Error Found.
This document validates as CSS level 3 !
Warnings (1)
URI : navBar.css
9
Imported style sheets are not checked in direct input and file upload modes

productDetails.css

Secure | https://jigsaw.w3.org/css-validator/validator

The W3C CSS Validation Service

W3C CSS Validator results for productDetails.css (CSS level 3)

Jump to: Warnings (4)

W3C CSS Validator results for productDetails.css (CSS level 3)

Congratulations! No Error Found.

This document validates as CSS level 3 !

Warnings (4)

URI : productDetails.css	
129	Property -webkit-transform is an unknown vendor extension
173	Property -webkit-transition is an unknown vendor extension
174	Property -moz-transition is an unknown vendor extension
175	Property -o-transition is an unknown vendor extension

productPage.css

Secure | https://jigsaw.w3.org/css-validator/validator

The W3C CSS Validation Service

W3C CSS Validator results for productPage.css (CSS level 3)

Jump to: Warnings (3)

W3C CSS Validator results for productPage.css (CSS level 3)

Congratulations! No Error Found.

This document validates as CSS level 3 !

Warnings (3)

URI : productPage.css	
74	Property -webkit-transition is an unknown vendor extension
75	Property -moz-transition is an unknown vendor extension
76	Property -o-transition is an unknown vendor extension

shoppingCart.css

Secure | https://jigsaw.w3.org/css-validator/validator

The W3C CSS Validation Service

W3C CSS Validator results for shoppingCart.css (CSS level 3)

Jump to: Warnings (1)

W3C CSS Validator results for shoppingCart.css (CSS level 3)

Congratulations! No Error Found.

This document validates as CSS level 3 !

Warnings (1)

URI : shoppingCart.css	
96	Property -webkit-transform is an unknown vendor extension

References

Bazon, M. (2012) *UglifyJS — JavaScript parser, compressor, minifier written in JS*. Available at: http://lisperator.net/uglifyjs/ (Accessed: 10 January 2017).

CES (2007) *CES* Available at: http://www.ces.tech/ (Accessed: 7 January 2017).

Compressing your JavaScript with closure compiler (2015) Available at: https://developers.google.com/speed/articles/compressing-javascript (Accessed: 10 January 2017).

mrdoob (2017) *Mrdoob/three.Js*. Available at: https://github.com/mrdoob/three.js (Accessed: 20 December 2016).

The history of the web - W3C Wiki (2014) Available at: https://www.w3.org/wiki/The_history_of_the_Web (Accessed: 9 January 2017).

travelandleisure.com (no date) Available at: http://cdn-image.travelandleisure.com/sites/default/files/styles/1600x1000/public/shanghai0515-cityscape.jpg?itok=I8_OMesH (Accessed: 7 January 2017).

Twitter (2016) *Embedded Timelines — Twitter developers*. Available at: https://dev.twitter.com/web/embedded-timelines (Accessed: 7 January 2017).

w3schools.com (2016) *JavaScript best practices*. Available at: http://www.w3schools.com/js/js_best_practices.asp (Accessed: 28 December 2016).

3D models for free (2017) Available at: http://tf3dm.com/ (Accessed: 20 December 2016).

TurboSquid (2017) 3D models for professionals: TurboSquid. Available at: http://www.turbosquid.com/ (Accessed: 20 December 2016).