2.4 Prototyping Notes

Jacob Cheatley 2015

This document details the components, tools and techniques which I have explored during my time creating a website project for 12IA.

# End User

This section is about what platforms I will need to support – what will the majority of users be using in terms of browser and operating system, screen resolution, what content will they be able to load, and also how to make the user experience as good as possible - employing web design principles to make my site as ‘good looking’ and user friendly as possible.

## Browsers:

### Browser usage statistics:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Browser | StatCounter | W3Counter | Wikimedia | Netapplications |
| Chrome | 51.8% | 42.7% | 47.9% | 20.6% |
| Internet Explorer | 21.7% | 15.8% | 18.3% | 58.9% |
| Firefox | 18.5% | 16.4% | 17.1% | 13.3% |
| Safari | 5.0% | 15.0% | 4.4% | 5.9% |
| Opera | 1.5% | 3.3% | 1.5% | 0.9% |
| Other | 1.6% | 6.8% | 10.8% | 0.5% |

Source: <http://en.wikipedia.org/wiki/Usage_share_of_web_browsers#Summary_table>

This table show the relative traffic tracked by four different tracking tools of the five major browsers (and other).

Each tracker will have its own biases towards certain browsers, as StatCounter for example uses a different tracking method to NetApplications – page views versus unique users, with StatCounter’s criticisms of NetApplications on this matter summarised here: <http://www.neowin.net/news/statcounter-critical-again-of-microsofts-ie-browser-claims>, noting not only tracking method (which heavily overvalues countries such as China with large populations, but little usage time), lower volume of websites tracked but also pay to use nature encouraging only certain types of businesses to use the NetApplications tool. Personally I think NetApplications results are much more heavily skewed.

In addition, trackers such as W3Counter are likely to be biased towards Safari, as a very sizeable portion of web developers (and other creative IT stuff), which is the audience of W3 websites, are using Mac. For a similar reason, IE will be under represented, as basically no web developer ever wants to use IE for anything other than testing their website and trying to get it work on the thing.

Another thing to take into consideration is not just the global average audience, but MY audience for this project. If the project is for the class for example, I highly doubt many people will be using browsers other than Chrome or Firefox, and I very much doubt (and hope) no one is using Internet Explorer. As a result I will focus mainly on supporting fully Chrome and Firefox. If my target audience is more likely to be using other browsers, then I should instead be focussing mainly on supporting those. Of course, support for all major browsers should be attainable and so browser usage statistics of my targeted audience should only be used for prioritisation of browser support rather than exclusive browser support to the majority of users.

## Screen Resolutions:

### Screen Resolution Usage Statistics (as of January 2014):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Other high | 1920x1080 | 1366x768 | 1280x1024 | 1280x800 | 1024x768 | 800x600 | Lower |
| 34% | 13% | 31% | 8% | 7% | 6% | 0.5% | 0.5% |

Source: <http://www.w3schools.com/browsers/browsers_display.asp>

In a similar way to how W3 was possibly biased towards certain browsers, it is likely they are also biased towards certain resolutions, as most web developers aren’t going to be using very low resolution monitors (they may mess around with screen resolution to test, but this likely will not be during visits to W3). However, again this is dependent on the targeted audience, and the most useful statistics to me on what to prioritise are going to be the statistics of my audience.

Also, supporting phones is another issue I will likely have to take care of. With such a large variety in common resolutions (from 240p all the way to 1080p), and an equally large variety in pixel density (from 120dpi to 320dpi), a lot of effort will need to go into properly scaling the website for screens of these sizes. This means I will need to use responsive web design to dynamically adjust based on the user’s resolution and device.

## Fonts:

### Web Safe Fonts:

<http://www.w3schools.com/cssref/css_websafe_fonts.asp>

Using these web safe fonts (and the techniques detailed in the page) will make sure that I have as high a level of font compatibility for users without them having to download some new font to view my website properly.

### Google Fonts:

<https://www.google.com/fonts/>

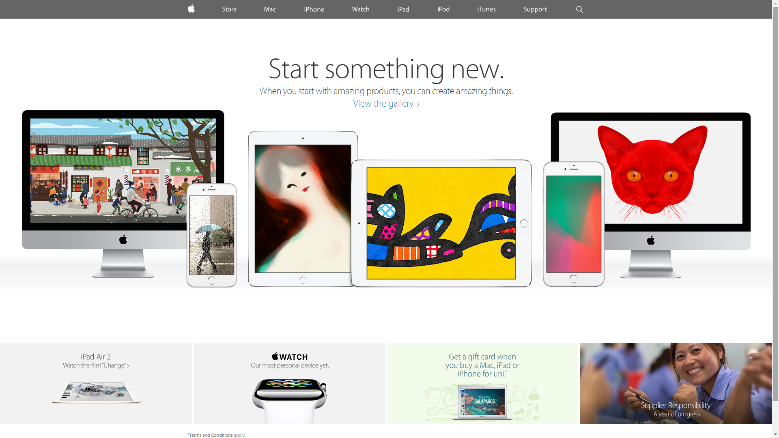
Google Fonts is a massive repository of fonts in a wide variety of styles which can be easily used and will just be fetched from the Google servers when a user loads a page. This will have an impact on page load times, which may be needed to be taken into consideration, but the gain of a more fitting font could be massively positive.

## Web Design:

### Websites Which Show Web Design Principles:

#### Simple and Clear…

<https://www.apple.com/nz/>



Apple is famed for simple design in its products, and its website is no exception - spacious, white and logically ordered to make it easy to find relevant information and have a sense of professionalism.

#### …Confused and Cluttered

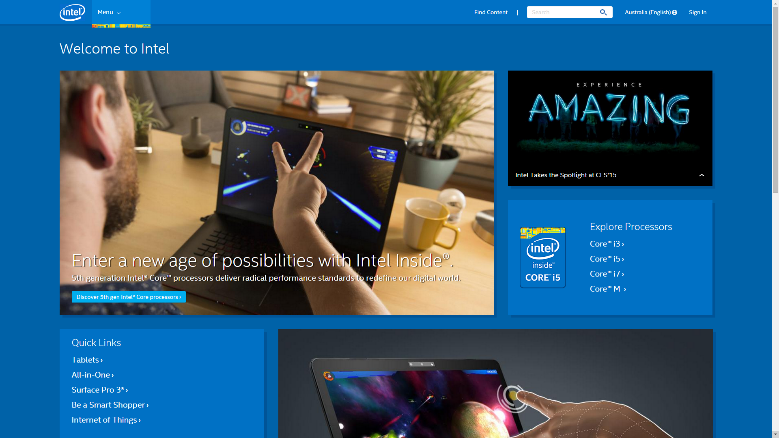
<http://www.lingscars.com/>



I don’t think I have to explain why this website is bad. It just has far too much random stuff, stuff playing in the background, ugly stuff, moving stuff, and it’s all one massive page full of advertising. This website is a far cry from Apple’s simplicity; it will drive customers away with its unfocussed web page.

#### Good Colour Scheme…

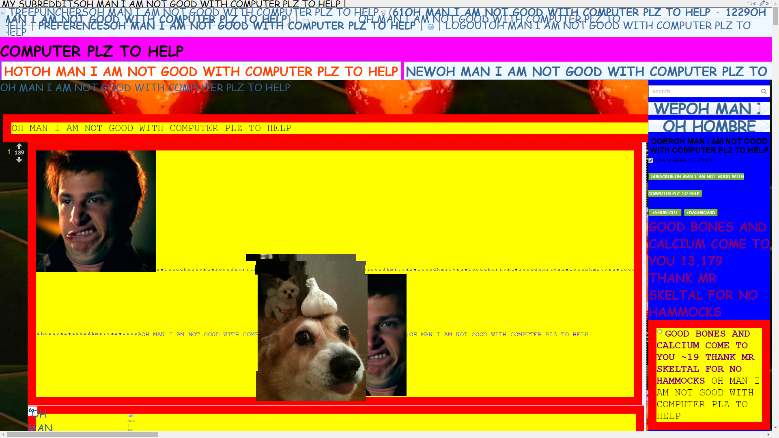
<http://www.intel.com/>



Intel uses a simple, cohesive colour scheme with slightly different shades of blue for emphasis. It is focussed on their brand, and is very easy of the eye while still standing out from the many white websites in the web.

#### …Poor Colour Scheme

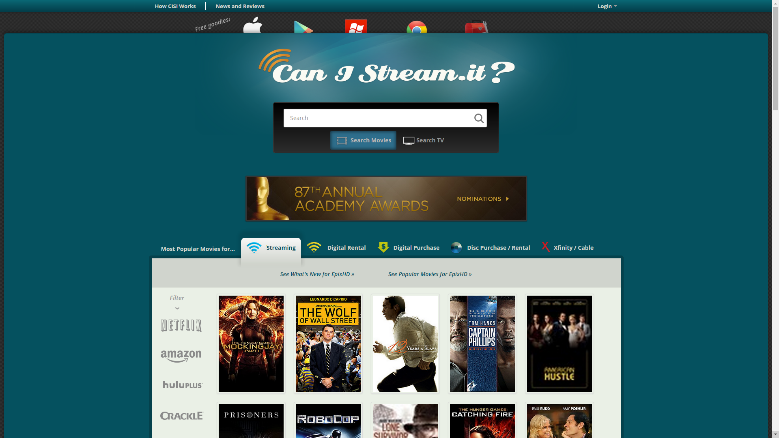
<http://www.reddit.com/r/ooer>



Well, clearly /r/ooer has some other SERIOUS issues, but its colour scheme is something that REALLY stands out as hilariously bad. Clearly this subreddit is designed to look this terrible, but honestly it is quite representative of sites which beginners make colour scheme wise. Every colour here has only F’s and 0’s in its hex code. It’s disgustingly terrible.

#### Visually Appealing Backgrounds…

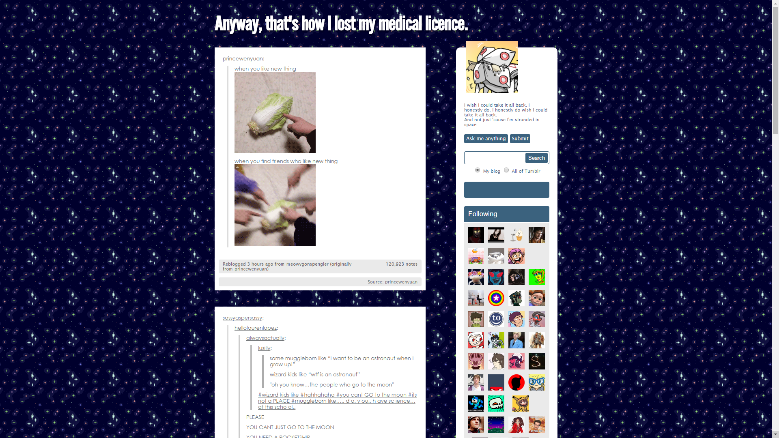
<http://www.canistream.it/>



Can I stream it uses a striking, but not painful solid turquoise (I think…) background around its main content which keeps your eyes on the page without it being terrible to look at. This is wrapped in a subtle black and grey grain tile, which makes the site seem professional, and consistent.

#### …Visually Appalling Backgrounds

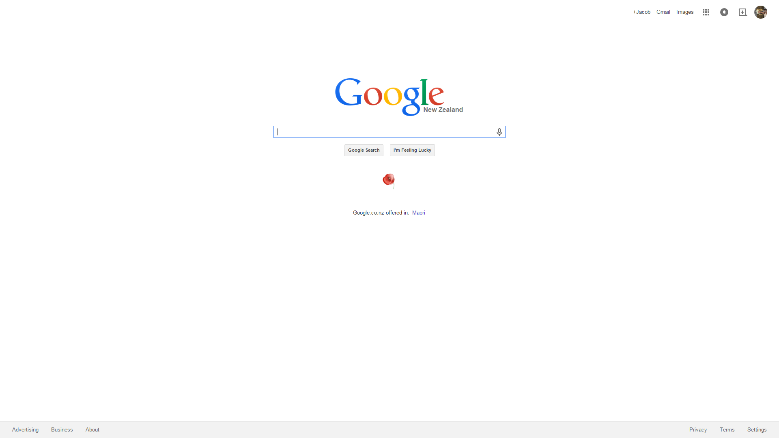
<http://vineweaver.tumblr.com/>



Sorry Ben, but this site’s tiling background is painful. Colours too contrasting for a start, and the tiling effect is incredibly obvious. This distracts the user from the content on the page when a background should instead reinforce the content and draw your eye to it. A more subtle approach is generally needed for this.

#### Highly Functional…

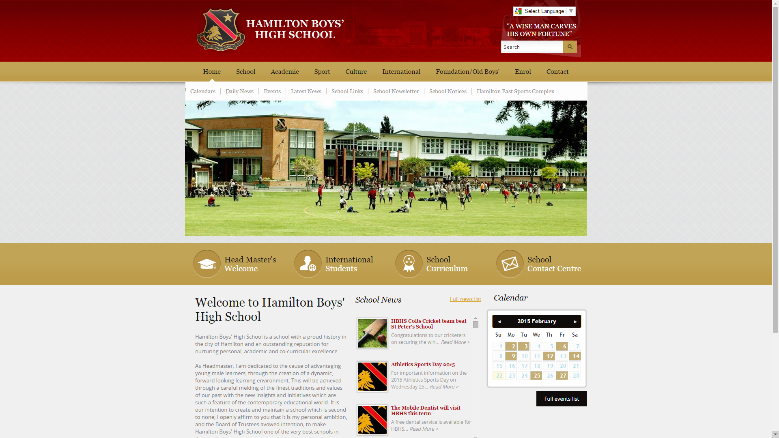
<https://www.google.co.nz/>



Google search does its function well. The page is focussed on this function, as a user types results automatically pop up, related searches are suggested and on top of this the search algorithm is awesome. All of these things combine to make Google simple to use and HIGHLY functional.

#### …Highly Disfunctional

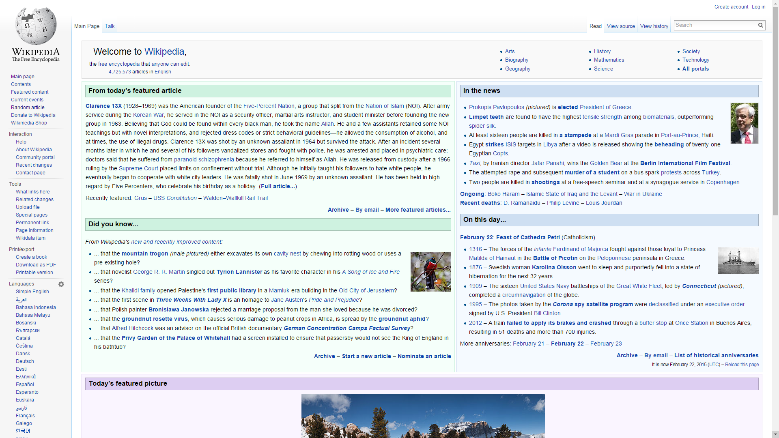
<http://www.hbhs.school.nz/>



The main issue here is navigation, primarily the top navigation bar. Sometimes multi-level, sometimes not, always inconsistent and requires additional clicks. Oh and dodging your mouse around mouse over parts of the submenu in order to click on other things. It’s a simple thing, but the navigation function of the HBHS is frustrating for a user to use. Of course, it’s hardly fair to compare a school website to Google in terms of functionality.

#### Awesome Content…

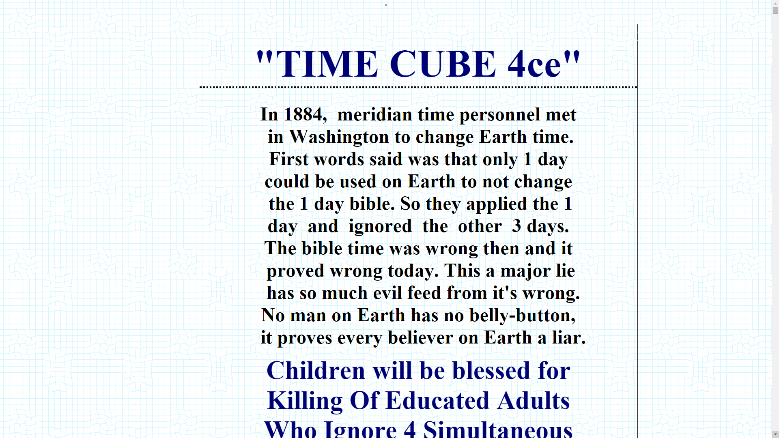
<http://en.wikipedia.org/wiki/Main_Page>



There is more to a website than its look, content is incredibly important. Wikipedia is the home of a massive amount of easily searchable knowledge. Although it is editable by everyone and occasionally things slip through, it is generally well sourced, well moderated and provides a very useful online encyclopaedia utility. It is the epitome of awesome content, with facts and figures on everything.

#### …Crazy Content

<http://www.timecube.com/>



Try to manage to read through the massive font on this website without having a massive brain aneurysm. You will have trouble.

### Web Design Do’s and Don’ts:

|  |  |
| --- | --- |
| DO: | DON’T: |
| The opposite of everything on the don’t list. | Have neon bright highly contrasting colour schemes  Have obvious tiling backgrounds  Have inconsistent page to page navigation  Have loads of animated gifs  Have a cluttered and disorganised page  Have auto-playing sounds  Have misaligned boxes strewn about the place  Have irrelevant ads across the page  Have endless walls of text  Have giant or tiny text  Have too much content on a page to load quickly  Have marquee scrolling text  Have out of order heading hierarchy  Have a visit counter  Have blinking or flashing text – or anything  Have users need to download a plug-in  Have useless splash pages  Have popups  Have all your text centred |

### Trends in Web Design:

#### Navigation:

In the beginning of the internet, almost every site used right aligned columns with lines and lines of navigation. More recently, horizontal navigation bars with drop downs, and pop down vertical navigation bars have been used more commonly, with horizontal navigation used for websites on desktops with higher resolution, and easier ways to scroll to the top of pages. Horizontal navigation bars are generally used on desktop websites as they are not going to cut into the content on the page, they waste less space on a page above the text than besides it, where, in addition to context menus, cramped up text will likely occur.

Vertical navigation bars which pop down after a button (which follows the view) is clicked, or, more likely tapped, are used commonly on mobile or tablets where screen real estate is at a much higher premium, and the lack of scroll wheel or mouse control makes it inconvenient to have to go to a specific place on a page in order to navigate away from it.

In addition, websites will use contextual menus on each page to allow the user to navigate to related pages. These are generally on one of the sides.

#### Backgrounds:

There are three main types of backgrounds – solid colour, tiled and large images.

Solid colour backgrounds have the benefit of clarity and simplicity – a plain white or other light colour portrays a clean and clear look, and it is generally accompanied by simple box style web design (like Windows 8 Metro, or Google’s Material Design). Another benefit of a solid colour is the near zero loading time required.

A tiled background, if done properly, has many of the same benefits and works in much the same way as a solid colour background, but it can provide a subtle texture which can enhance the feel of a website – for example a weaving craft shop website could use some kind of tiling woven background, or a farming website could use something that looks a bit dirty or grassy. Done correctly this is a great way to subtly give the user information about your brand and be aesthetically pleasing.

Large images (or videos), which can often fill up the entire page, are a very recent trend as broadband speeds have gotten much better over the years. They can be very aesthetically pleasing and give loads of information – they effectively are full screen videos with text overlaid and menus above or below.

### Websites for Web Design and Development:

<http://www.reddit.com/r/webdev/> - Some web development stuff

<http://www.reddit.com/r/web_design/> - Some web design stuff

<https://stackoverflow.com/> - The home of all programming help, going to be especially useful for JavaScript

# Components and Materials:

This section details and compares types of media – components and materials which I can integrate into my website. The main aim of comparison here is to decide the optimum file type to use for a specific task as a result of my research.

## Images:

“A picture paints a thousand words.” Graphical content is extremely important for conveying information and making a website look good. I will certainly use various images, both created and photographed throughout in order to enhance my website.

### Image Type Comparisons

|  |  |  |  |
| --- | --- | --- | --- |
| Image Type | Background | Advantages | Disadvantages |
| GIF – Graphics Interchange Format  http://upload.wikimedia.org/wikipedia/commons/2/2c/Rotating_earth_%28large%29.gif | Created by CompuServe in 1987 for colour images on the internet.  Pronounced “GIFF” as in gift, NOT like how the creator claims, “JIFF” as in the cleaning product or the peanut butter. | Able to be animated.  Fairly small file sizes.  Lossless compression.  Allows transparency (as a colour, so no alpha).  Every major browser supports GIF.  Generally good for line drawings and simple images (but PNG is probably still better at this anyway). | Only 256 colours – can cause blocky or pixelated look.  Dithering stops high compression levels.  Even its main advantage – animation can be done with lower file sizes, higher quality and sound with HTML5 videos. |
| PNG-24 – Portable Network Graphic  http://upload.wikimedia.org/wikipedia/commons/4/47/PNG_transparency_demonstration_1.png | Created in 1996 as an open source alternative to GIF, without using the (at the time) patented LZW data compression algorithm. | Supports alpha transparency.  Lossless compression.  16.7 million colours.  Generally good for text, graphics, line drawings and where transparency is needed.  Every major browser supports PNG. | Relatively higher file sizes than JPEG for images such as photos. |
| PNG-8 – Portable Network Graphic  http://personabledesign.com/wp-content/uploads/2011/10/fworks2.jpg |  | Possibly slightly smaller file sizes than PNG-24.  Supported by IE6, but well, that means you are trying to support IE6. | Same as PNG-24, except only 256 colours. |
| JPEG – Joint Photographic Experts Group  http://upload.wikimedia.org/wikipedia/commons/e/e9/Felis_silvestris_silvestris_small_gradual_decrease_of_quality.png | Created by the Joint Photographic Experts Group in 1992 as a standard for how an image is compressed and decompressed. | 16.7 million colours.  Generally good for photos or art that’s semi-realistic.  Every major browser supports JPEG. | Lossy compression, and suffers from generational degradation (repeated saving lowers quality).  No transparency.  Looks terrible if you try to make a non-photographic image a JPEG due to high compression. |
| TIFF – Tagged Image File Format  I could put nearly image in here and it would make sense as a tiff. Also Google Image Search doesn’t support searching for a tiff, so here’s a picture of a cute puppy (which isn’t actually a tiff).  http://i.imgur.com/UZx3tUf.jpg | Created by Aldus for desktop publishing (a precursor to PDF) in 1986, mostly as a common format for scanners.  Actually started as a binary (black and white) image format, far from what it is now supporting trillions of colours.  Acquired by Abode in 1992. | Lossless compression, but can be saved compressed or uncompressed.  Supports multiple pages, even weird file tree type things. Basically several images and several versions of each image.  Supports layers.  Can store captions associated with images.  Supports a stupid number of colours, 16 bit per channel. (PNG-48 can do this as well). | Large file sizes, especially for photos.  Not natively supported by that many web browsers – including Google Chrome, Mozilla Firefox and Opera. Requires user extensions. |
| SVG  File:Vector-based example.svg | Open standard created by W3C (World Wide Web Consortium, same group as creators of HTML, CSS and XML) to be a web vector image format in 1999. | Able to be searched, indexed, **scripted** and compressed due them being defined in XML.  Vector graphics – not pixels, but rather lines, curves and shapes enabling infinite scalability without loss of quality.  Lossless compression.  16.7 million colours. | No major browser fully supports SVGs natively. They do mostly work however. |

## Video:

Although video content would be useful for a debating website, unfortunately I have been unable to get any of this content from our school in the time allowed, so I won’t be able to create video content.

### Video Type Comparisons:

|  |  |  |  |
| --- | --- | --- | --- |
| Video Type | Background | Advantages | Disadvantages |
| MP4 |  |  |  |
| WMV |  |  |  |
| AVI |  |  |  |
| GIFV/HTML5 video/WEBM |  |  |  |
| MKV |  |  |  |

## Code:

It would be rather difficult to create a website without creating some HTML, and it would also be very difficult to make it look any good or have all the bells and whistles without some CSS and JavaScript.

### Code Information:

|  |  |  |
| --- | --- | --- |
| Code Type | Background | Function |
| HTML |  |  |
| CSS |  |  |
| JavaScript |  |  |
| SCSS |  |  |
| Jade |  |  |
| jQuery |  |  |
|  |  |  |

# Tools

This section is about the tools, or programs which I could use to create my content. Although making websites can be done with very basic tools, exploring more featured options is a great thing to do.

## Text Editors:

Text editors, as their name implies, are used for editing text. Most commonly in this project, that will mean .html, .css, .js and .svg files. They may have some features such as basic auto completion, highlighting, or macros, but will otherwise be very limited and not run whatever you write inside them.

* Notepad

<image>

Notepad is installed on every Windows machine, so it’s easily usable. However it has no functionality besides very basic text manipulation, which is not very conducive to writing any kind of code. I will certainly not be using Notepad for this project (or at all).

* Notepad++ - <http://notepad-plus-plus.org/>

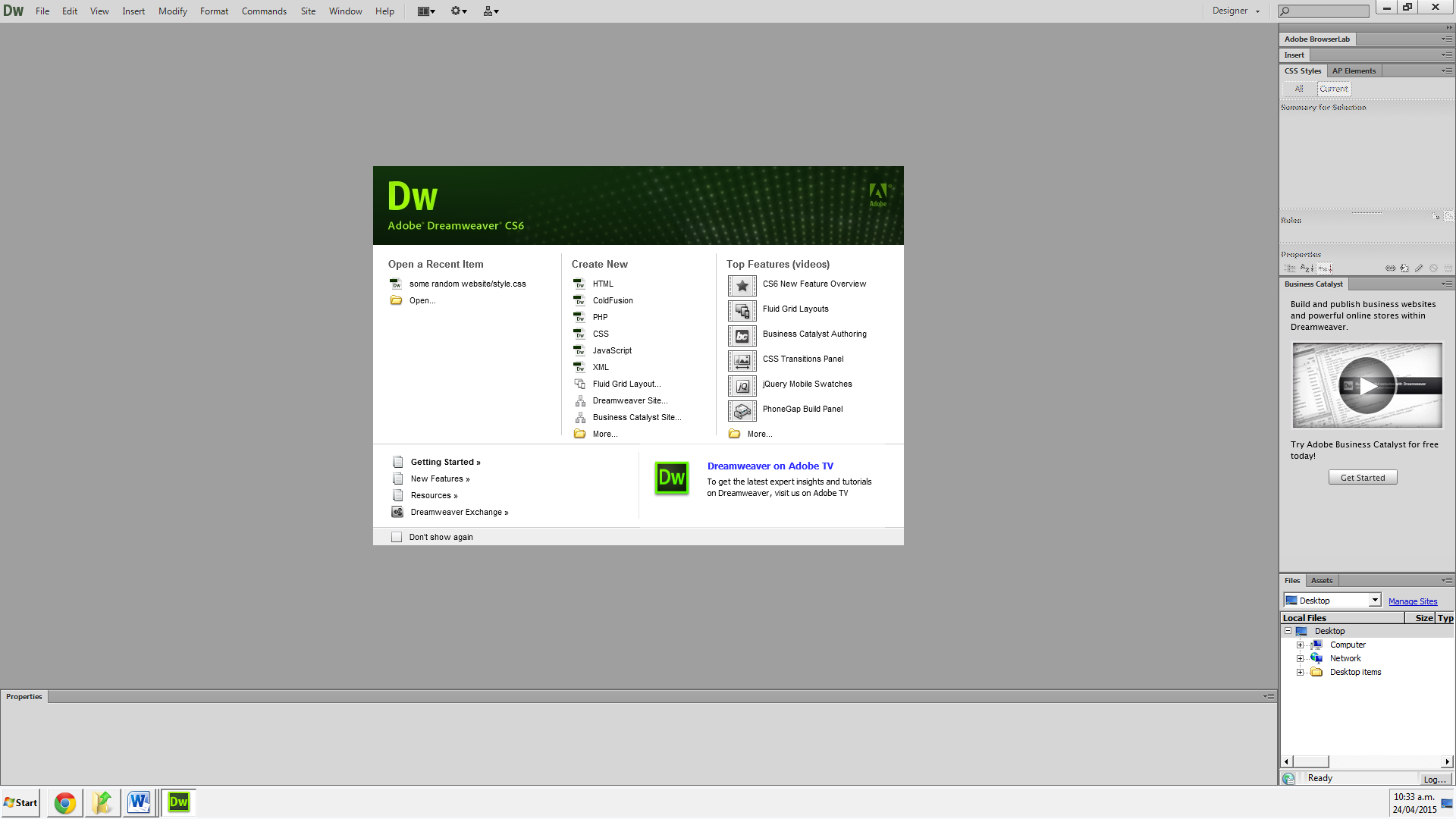
<image>

If Notepad is a knife, Notepad++ is a Swiss Army Knife (not quite on the level of Vim which is more like the Swiss Army itself, but I can’t really command an army). It has quite a few features for general code manipulation, including syntax highlight, some suggestions, and abilities to run stuff from it directly to browser. It is a general purpose tool which I may use, although sparingly, as dedicated IDEs (such as WebStorm) will be much more powerful (and are actually necessary for my purposes).

## IDEs:

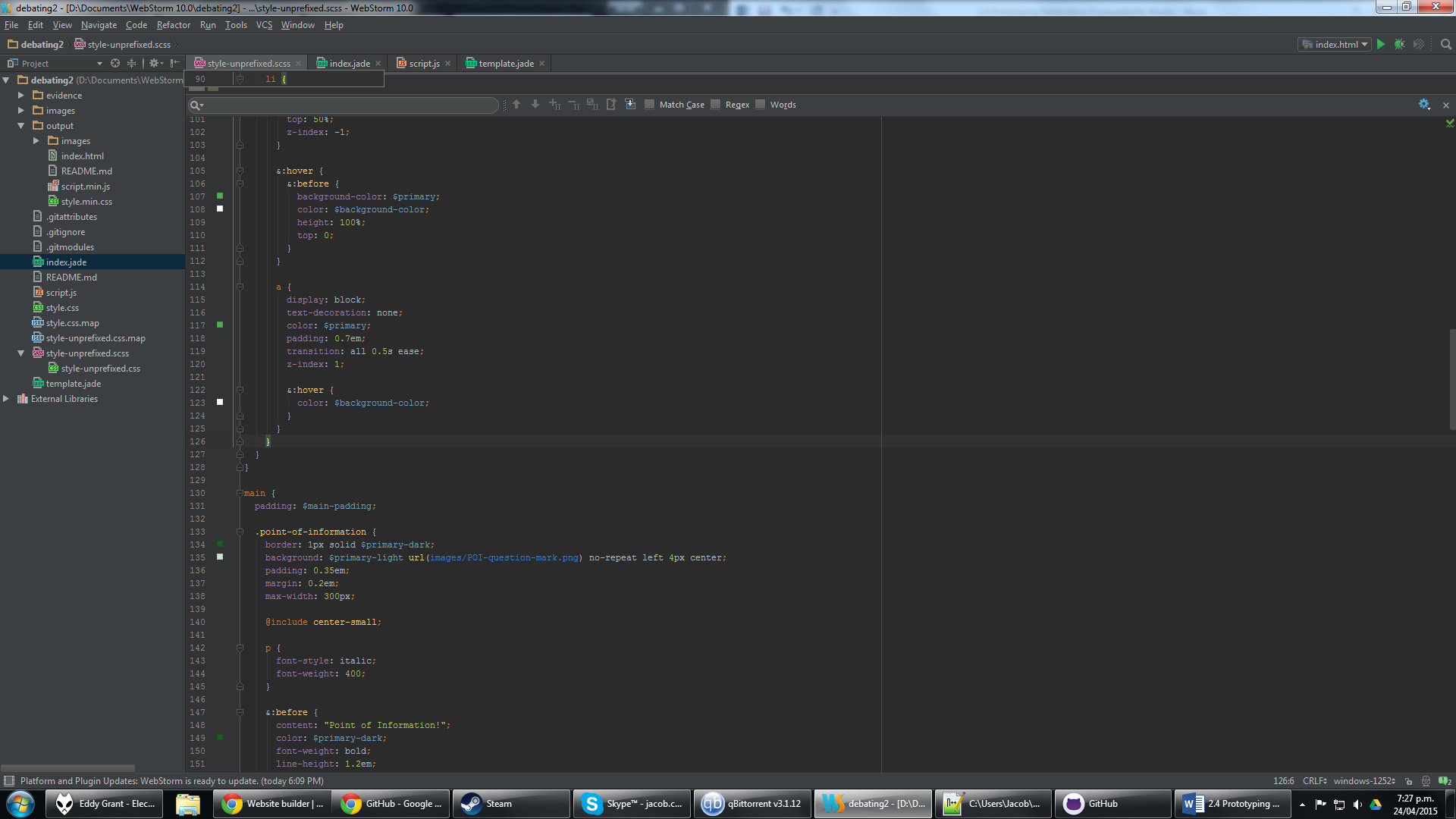
Integrated Development Environments are a text editor, a debugger, a compiler, a content manager and more. They are generally suited for a small range of purposes or languages.

* Dreamweaver - <http://www.adobe.com/nz/products/dreamweaver.html>



Dreamweaver is not something I have had much experience with. It has a WYSIWYG mode, which would be nice if it weren’t terrible. It lack features such as File Watchers which I certainly need, and I am more familiar with JetBrains style IDEs than any Adobe Product (as I use PyCharm for Python), so I think WebStorm is a better option.

* WebStorm - <https://www.jetbrains.com/webstorm/>



WebStorm is an IDE specifically for web based languages – so HTML, CSS, JavaScript etc. It is very feature rich and allows rapid editing with great auto-completion, and has other essential features such as its file watcher system and debugging tools. I have decided to use WebStorm for this project as my main editor of HTML, CSS and JavaScript (actually Jade, SCSS and jQuery) because of these reasons.

## Image Editors:

Image editors are used primarily for editing raster graphics, including but not limited to photographs, pixel art and digital art.

* Adobe Photoshop - <http://www.adobe.com/nz/products/photoshop.html>

<image>

Adobe Photoshop is the industry standard for photo manipulation and raster graphics for a reason. It has a vast array of tools and features, along with great tutorial support. I didn’t like it initially because it felt somewhat bloated and required too much to learn to do things which I found simple in Paint.NET. However, after some further use I saw the Adobe Light(room) and I will now use it as my primary image editor, simply because of how much it can do.

* Paint.NET - <http://www.getpaint.net/index.html>

<image>

Paint.NET used to be one of my favourite image editors until I actually got my way around Photoshop. It is very simple, yet still has many of the features that are essential for photo manipulation and graphics creation. However, it is lacking a lot of the things which I have found to be great in Photoshop, and so I likely won’t use it except for a few things, mostly pixel art (where you don’t need such a large array of features), although I don’t think I will be making any pixel art for this project.

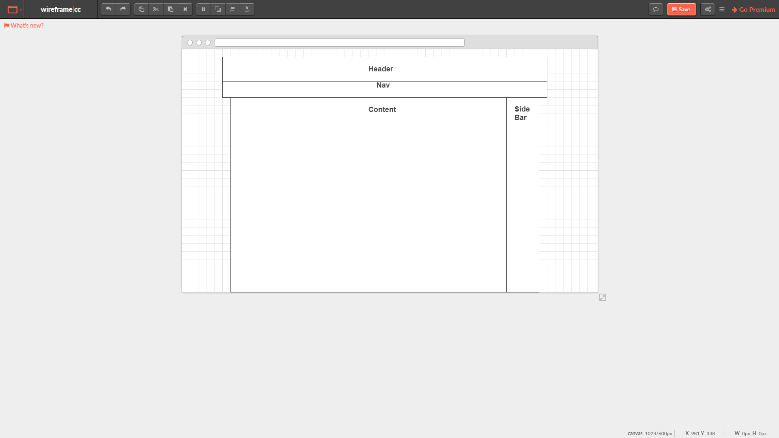
## Website Wireframing Programs:

These are designed to create wireframes – basic blocked out box models for many projects, and have some additional features such as text. They are not so much designed to figure out colour schemes and visuals, but rather content placement, alignment and functionality.

* Pen and Paper

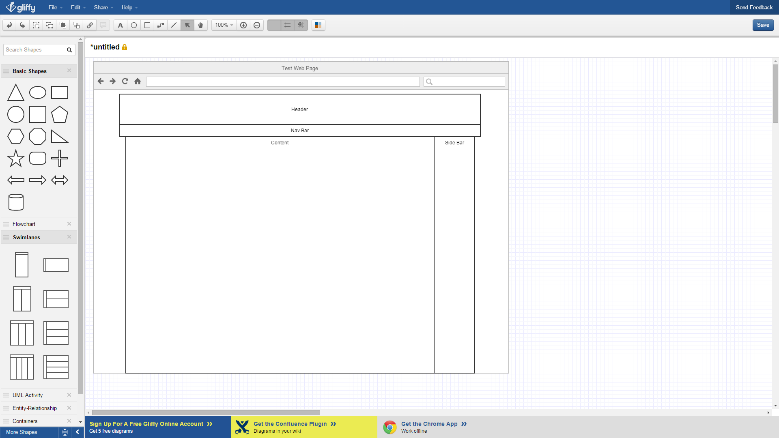
Well, it’s always an option, and it’s the one I’m going to do because it’s fast and extremely intuitive (requires no learning at all). Also because the Wireframe stage of the website is a brief and simple one so I don’t want to spend time learning something which won’t get much return.

* Wireframe - <https://wireframe.cc/>



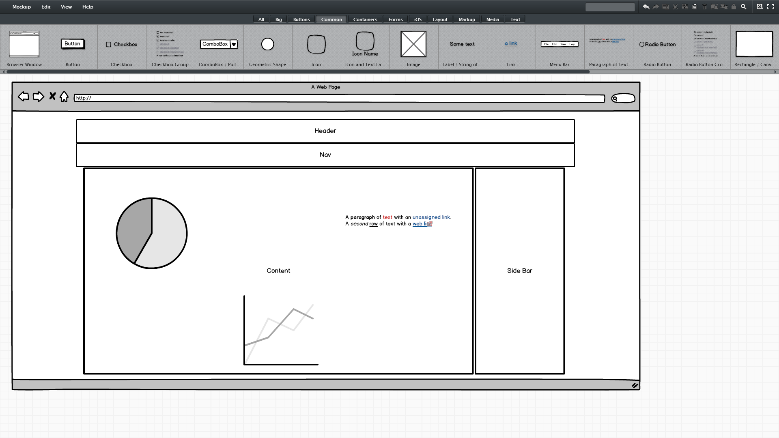
I found Wireframe generally frustrating to use, primarily because of the generic rectangle for every shape, and odd snapping behaviours, as well as a general lack of very basic features. I would rather use crayons on corrugated iron.

* Gliffy - <https://www.gliffy.com/>



Gliffy was an improvement on Wireframe. It had a larger range of shapes, the ability to have text as part of a shape, and a much more featured interface, requiring less clicks to do more. It is much more functional. I would still probably rather use pen and paper in very early stages, for speed alone.

* Balsamiq - <https://balsamiq.com/>



Balsamiq is feature rich, easy to use (mostly, some precision needs to be learned), and intuitive. It has loads of different components which you can actually add your own mark-up to in order to customize their appearance and text displayed etc. Even this free web demo version in the few minutes I spent with it is very useable, for not just simple wireframing, but mock-ups too. I would still probably rather use pen and paper in very early stages, for speed alone.

## Hosting/Version Control/Source Management:

* GitHub - <https://github.com/>

<image>

GitHub is a rather complicated but very powerful service. It is designed for open source collaboration on projects primarily, which I clearly won’t be doing (as this is a personal project), however it also has key features I need. Firstly version control, to be able to easily revert to a previous commit, and secondly (and perhaps more importantly) it is also a good hosting service that creates readable, custom URLs (i.e. not jumbles of characters like a Google URL shortener would do).

# Techniques

This section is dedicated to exploring the uses of tools that I will be using, and comparing them to decide which is best.

## Image Editor Techniques:

### Adobe Photoshop Techniques:

|  |  |  |
| --- | --- | --- |
| Technique | Description | Screenshot Example |
| Selection Tool | Provides a variety of different shapes and types of selections, which can then be further manipulated by other tools. |  |
| Magic Wand Selection Tool | Selects based on colours of the image, can be contiguous or global. Very useful for selecting backgrounds. |  |
| Brush | Brush with soft colours, functions how you would expect a paint brush in real life to work.  Has anti-aliasing, is semi-transparent. |  |
| Pencil Tool | Draws aliased hard colours. Especially useful for pixel art. |  |
| Eraser Tool | Deletes parts of an image.  Photoshop settings didn’t like to show actual transparency here for some reason. |  |
| Move Tool | Moves the current selection.  Showing Transform Controls allows all kinds of transformations to be applied to the current select, e.g. stretch, scale, shear, rotation. |  |
| Crop Tool | Used to take subsections of an image and remove what is outside of that area. |  |
| Eye Dropper Tool | Picks a colour from the current image which will become your primary selected colour for brushes, fills and pencils etc. |  |
| Save to Web | Used to preview and create compressed versions of an image, optimised for web, in various file types. |  |
| Spot Healing Tool | Used ~~by models~~ for fixing small imperfections by sampling an area around the brushed selection. | Before:    After: |
| Filter Menu | Provides a large variety of different effects which can be applied to the selection. For artistic – distorting, stylizing, pixelating etc., or for practical – blurring, sharpening etc. |  |
| Filter: Blur Submenu | A variety of types of blurs, for blending, adding the illusion of motion or even reducing file sizes in some circumstances. | Gaussian Blur Example: |
| Filter: Distort Submenu | Many effects which can dramatically change the shape of an image, bending and shifting things around. | ZigZag Example: |
| Filter: Render Submenu | Generates specific images or patterns, like clouds or lens flares. | Lens Flare Example: |
| Filter: Stylize Submenu | Various filters which can heavily modify the look of an image. | Find Edge Example: |
| Paint Bucket Tool | Fills an area of similar colour in the current selection with the primary colour. |  |
| Gradient Tool | Fills the current selection with a gradual blend from one colour to another, in a variety of shapes. |  |
| Layers | Layers are like sheets of paper which you can overlay on top of each other, to preserve previous work and apply effects, or easily arrange objects in a non-destructive way. |  |
| Shapes Tool | Basic primitive shapes which can be used to build up more complex objects, or just block stuff out. | Android created with many different shapes: |
| Layer Mask | Used to selectively block or show certain parts of the layer below, great for only taking certain parts of a gradient or for non-destructively removing a background. |  |
| Text Tool | Text. It is fairly self-explanatory. Can use a variety of fonts, colours, sizes etc., in addition to various effects to convey textual information. |  |

### Paint.NET Techniques:

|  |  |  |
| --- | --- | --- |
| Technique | Description | Screenshot Example |
| Selection Tool | Selects an area which can then be further manipulated. |  |
| Magic Wand Selection Tool | Selects an area based on similar colours. |  |
| Brush Tool | Brushes an area with a colour. Fade seems to work differently to the brush tool on Photoshop, Paint.NET does much less blending even with very low hardness. |  |
| Pencil Tool | Draws aliased single pixels. For larger sizes of aliased drawing, turning off antialiasing in brush tool works to function like a larger sized pencil would in Photoshop. |  |
| Eraser Tool | Removes whatever is erased, replacing it with transparency. |  |
| Move Tool | Manipulates the current selection’s position, rotation, scale etc. |  |
| ~~Crop Tool~~ Crop to Selection Menu | Removes any part of the image outside of the bounds of the selection. Unlike Photoshop, where you can un-crop, Paint.NET is destructive. |  |
| Eye Dropper Tool | Picks a colour for your primary or secondary colour from the image. |  |
| ~~Save to Web~~ Resize, Save As | Combination of these actions allows you to resize and change the quality of an image to get an appropriate size. Less customizable than Photoshop, also requires undoing these actions if you want to keep a full resolution and quality version of an image. |  |
| ~~Spot Healing Tool~~  Nothing similar without extensions |  |  |
| ~~Filter~~ Effects Menu | A variety of filters/effects which can be applied on the current layer of the image. |  |
| ~~Filter~~ Effects: Blurs Submenu | Blurs which can be used to provide illusions of motion, focus or reduce file size (in some file types). | Motion Blur Example: |
| ~~Filter~~ Effects: Distort Submenu | Many effects which can dramatically change the shape of an image, bending and shifting things around. | Crystalize Example: |
| ~~Filter~~ Effects: Render Submenu | Generates specific images or patterns, like clouds or fractals. | Clouds Example: |
| ~~Filter~~ Effects: Stylize Submenu | Effects which mostly focus on some form of edge detection method to change what the image looks like. | Edge Detect Example: |
| Paint Bucket Tool | Fills an area of the same colour within the current selection with another colour. |  |
| Gradient Tool | Fills an area with a gradient from one colour to another in a variety of shapes. |  |
| Layers | Layers are like sheets of paper which you can overlay on top of each other, to preserve previous work and apply effects, or easily arrange objects in a non-destructive way. |  |
| Shapes Tool | Create a variety of simple shapes. Unlike Photoshop they are instantly rasterized as soon as they are finished being placed, so cannot be edited after being created. |  |
| ~~Layer Mask~~  Nothing similar without extensions |  |  |
| Text Tool | Text – fonts, colours, sizes etc. |  |

## Wireframe Techniques:

### Pen and Paper Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Using a pen | I’m not seriously going to do this. Move on to the next section. | http://vignette3.wikia.nocookie.net/thefakegees/images/6/6d/Look_of_Disapproval.png/revision/latest?cb=20130120182037  It’s pen and paper. I am not doing techniques for pen and paper. |
| Using a ruler | … | … |

I didn’t explore other tools enough to provide useful content for their techniques, and I’m not using them.

## HTML Techniques:

### Vanilla HTML Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Linking to external scripts and stylesheets | Linking to an external stylesheet is essential to easily have consistent styling across all pages. |  |
| Using classes and IDs | Class and ID attributes can be applied to elements so they can be further manipulated by JavaScript and CSS. |  |
| Using HTML5 semantic tags | New in html5, semantic elements allow a search engine to know what exactly a ‘thing’ is instead of creating div soup on a page. This is also useful to quickly tell what exactly a thing is for a developer. |  |
| Moving the end of a closing tag to the next line to stop RANDOM STUPID SPACING between inline-block elements | HTML is rather finicky with inline blocks. The newlines between tags are treated as spaces, which I don’t want. So the tag is closed on the next line, so there is no space. |  |

### Jade Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Doing what you’d normally do in HTML |  |  |
| Attributes | You can easily apply attributes to elements with keyword arguments inside parentheses. |  |
| Extends | Inheritance from other .jade files is very powerful and allows sharing of mixins, JavaScript variables and most importantly Jade structure. |  |
| Mixins | Mixins are kind of like templates for a single element or group of elements. They can take parameters, so they act like shorthand functions for larger amounts of jade content. |  |
| Blocks | Mixins and files which are being extended can have blocks which can be overwritten. |  |
| JavaScript integration | Using –‘s at the beginnings of lines will denote the following as JavaScript which you can use for additional functionality. (This JavaScript will not be present in the page as a script, it can just be used to create Jade code before the Jade is transpiled to HTML) |  |

## CSS Techniques:

### Vanilla CSS3 Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| CSS selectors | Selecting an element to apply styling to is done using selectors. This is in the format of the name of the HTML tag, with spaces denoting elements within the previous element, and commas denoting applying styles to each comma separated selector. |  |
| Styling an element | Styling an element consists of a series of pairs of attributes and values for those attributes which will affect how the selected element looks. |  |
| Fall-back fonts | Multiple fonts can be used for a font-family attribute, meaning that if the browser cannot load the first font, it will “fall back” to the next in the list. |  |
| Transitions | Transitions are new to CSS3 and allow changes to an attributes values to have a transition with a set duration, delay, smoothing function etc for more attractive pages (especially useful for :hovers and similar). |  |
| RGBA | RGBA is an alternative way to declare colours instead of hexadecimal and allows an alpha channel value of 0-1 for transparency. |  |
| Pseudo-classes | Pseudo-classes modify selectors in order to specify the element’s state. |  |
| Pseudo-elements | Pseudo-elements are a few extra ‘fake’ elements which can be styled (e.g. creating an element before another element, or modifying the first letter of an element). |  |
| Importing | Using @import allows web hosted and local stylesheets to be imported into the stylesheet and used, e.g. for reset css files or Google fonts. |  |
| Media queries | Media queries can modify the styling of elements when the user’s device has different specifications, e.g. screen vs print, or certain screen sizes or aspect ratios. |  |
| Direct children | Allows selection of elements which are DIRECT children of an element, and not just contained somewhere within that element. |  |

### SCSS Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Variables | Variables for colours, fonts, width values etc. can be declared so those values be easily changed throughout the stylesheet. |  |
| Nesting | Nesting allows and easier shorthand and more logical – hierarchical method to apply styles to elements. |  |
| Mixins | A set of styles which can be easily applied across many elements and modified later. |  |
| Mathematics | Calculations can be done between variables and values before the page is transpiled to CSS. |  |
| Functions | SCSS has loads of built in functions, especially useful for color, string and number manipulation. |  |

## JavaScript Techniques:

### Vanilla JavaScript:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Creating variables | Variables. They do what you’d expect. They can be assigned to and read. |  |
| Creating arrays | Arrays are useful for storing and accessing many variables together. |  |
| If statements | If statements do a code block if they return true. |  |
| For loops | For loops are used for iteration with numbers. |  |
| Creating HTML elements | HTML elements can be created, and their attributes customized before being appended to the page. |  |
| Modifying HTML elements | HTML elements on the page can also have their attributes modified. |  |
| Functions | Functions are super useful for refactoring code to reduce repetition. |  |

### jQuery Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Checking if document is ready | This ensures that the page is fully loaded before the JavaScript runs so that you aren’t trying to access elements that do not exist. |  |
| Accessing an HTML element quickly by CSS selector format | jQuery constructors to select elements have a very similar syntax to CSS. From here methods can be called on them to modify their attributes, styling and content. |  |
| jQuery UI tooltips | jQuery UI is an extension of the jQuery library with many additional features for user interfaces. Tooltips are one such feature which provide hover – popups on elements based on their title attributes for additional information. |  |

## Other Tools Techniques:

### Webstorm Techniques:

|  |  |  |
| --- | --- | --- |
| TECHNIQUE | DESCRIPTION | SCREENSHOT EXAMPLE |
| Running a file | Files can be run directly from WebStorm which allows the use of it’s in browser debugging tools and is a handy shortcut. |  |
| Colour picker for css hex codes | There is a handy colour picker which will automatically create the hex code for a colour and provide a preview. |  |
| Preset colours | Additionally, HTML’s preset colours are easily accessible with auto completion help. |  |
| Shortcuts | Commenting out lines, running files, skipping around code blocks, selecting code blocks are all easily and quickly done with in editor keyboard shortcuts. |  |
| Code completion | Code completion provides a lot of context sensitive help, especially useful for JavaScript, but it also has functions in other supported languages. |  |
| File watchers | File watchers is a built in system to run external command lines with whatever arguments whenever a specific file is updated. This is useful for transpiling files from one type to another or using things such as AutoPrefixer to add vendor prefixes to CSS automagically. |  |

# The Website’s Journey

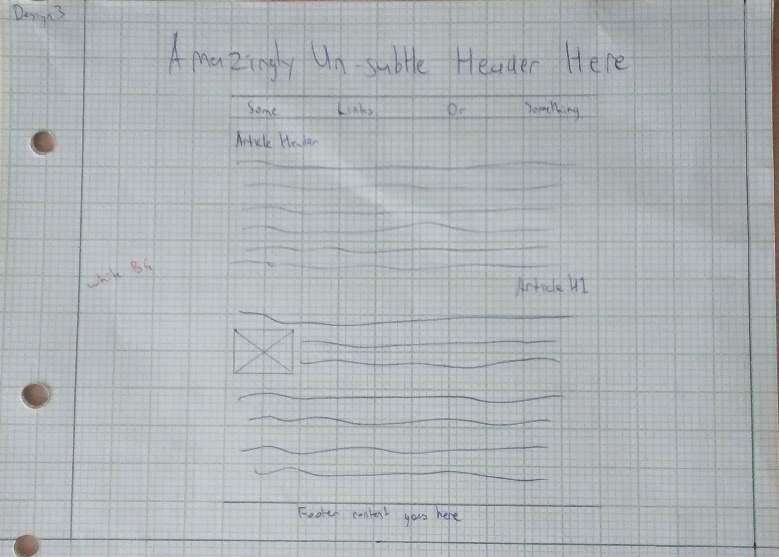
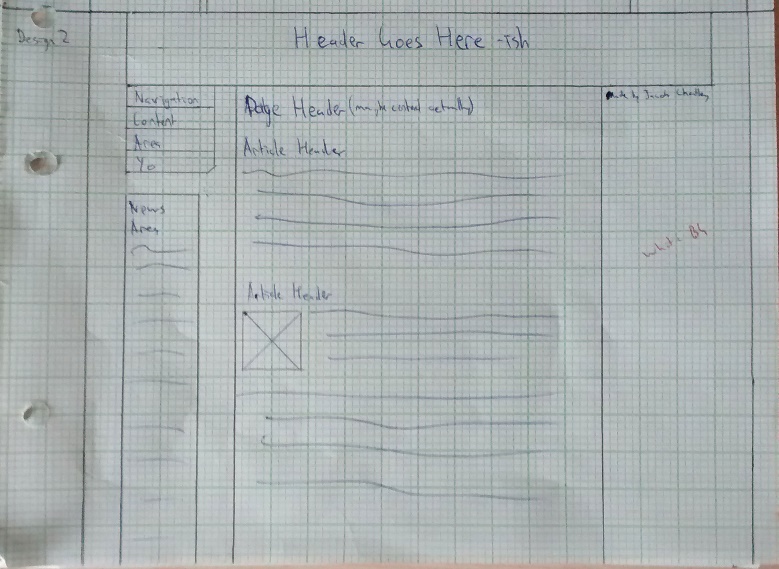
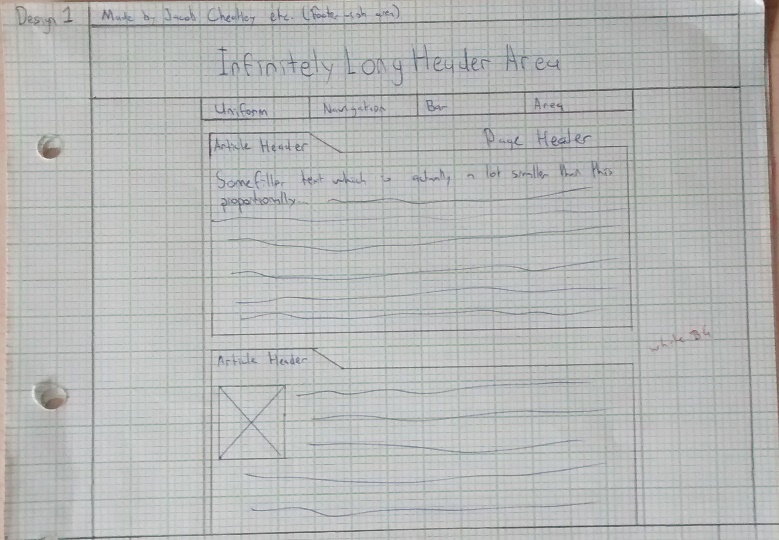
The section will be about gathering evidence of the creation of my website, to demonstrate usage of techniques and show the process of iterating on stakeholder feedback.

## Deciding on a Topic:

This was quite easy for me. The co-curricular activity I do which I have both done for the longest time and would be best for a website was debating. I will be making a website about what debating is, what to do in it and how to sign up, aimed at introducing people to debating at HBHS, especially at junior level.

## Initial Wireframing and Feedback:

I first created some basic wireframes using pen and paper, because I feel that it is faster, simpler and more portable to show off to people during school but outside of a 12IA period.



I got feedback from some people in the class:

### Jordyn Coxhead:

**Design 1**

The page header is not very obvious, articles should move down.

Has too much of a ‘blog page’ feel. There is a clear separation even though ideas will all be related.

**Design 2**

Move the ‘Made by Jacob Cheatley’ to the left, it is all by itself.

Likes the vertical nav.

**Design 3**

An off white background colour would be necessary.

Possibly a bit plain, although no real complaints.

### Blake Akapita:

**Design 1**

Favourite design, but the nav should be spaced out more from the surrounding elements.

**Design 2**

Vertical navigation is bad, should be horizontal instead.

**Design 3**

Not much to say about it.

### Cameron Salisbury:

**Design 1**

The folder design doesn’t make sense, it’s a web page not a filing cabinet.

Footer content at the very top of the page is not good.

Nav should be the same width as the header, not the content.

**Design 2**

Side navigation bars are terrible, as is a news section.

The rest of the design is standard and okay.

**Design 3**

Simple layout, but has potential to look best if done well.

### Isaac Poole:

**Design 1**

The angles next to article headers looks nice.

Header stretching across the page looks nice.

Good amount of white space.

Should move articles down and page header across.

**Design 2**

If news is present, have it on the right.

Nav on the left is bad.

Maybe have a gap between header and content.

**Design 3**

Nav should stretch to be the same size as the header.

### Alex Le Comte:

Alex is a debater, so one of my stakeholders in this project. He didn’t really give specific feedback, but he very much wants a “montage parody mode,” activated by a secret button. Here are some of his suggestions:

Dank memes

OG Kush

Snoop Dogg dancing with ‘Drop it Like it’s Hot’ playing in the background

Ca$h money (flashing gold all over the page)

References to Faze Clan, Optic and MLG

Deal with it glasses and people smoking blunts.

*I’m probably not going to do this.*

## Beginning Prototyping:

I have decided to skip a ‘mock-up’ phase. This is for a few reasons. Primarily because I don’t think it would add very much as I can more quickly and usefully iterate my actual prototype based on feedback than a mock-up – I wouldn’t find feedback on fonts and colours outside of the context of the actual website to be all that useful. Secondly because this would delay my ability to get this useful feedback, the opportunity cost is simply not worth it.

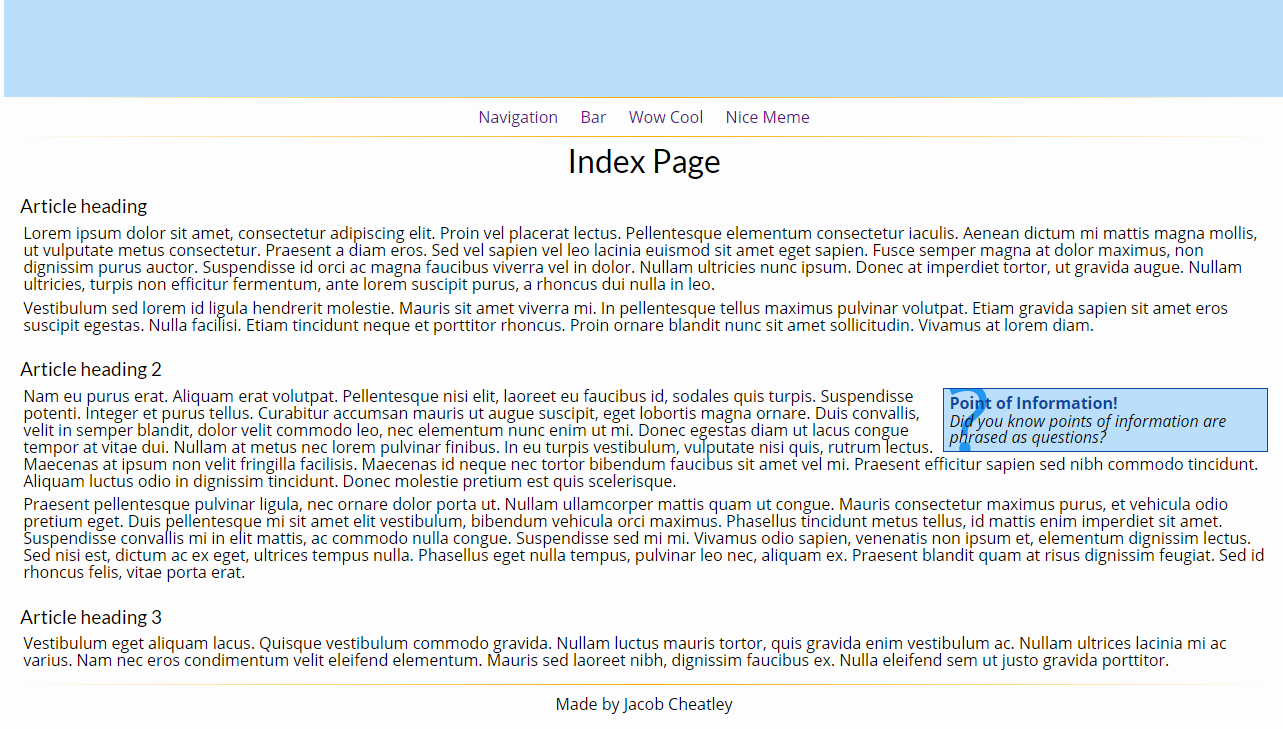
With this in mind, I began my prototype. I decided to use design 3, as no one really had any particular issues with it and I agree heavily with Cameron – that it has a lot of potential if done well. I also personally like it for the reason that is very open and spacious, no harsh borders and lots of white space.

After a few hours in WebStorm I came up with this:



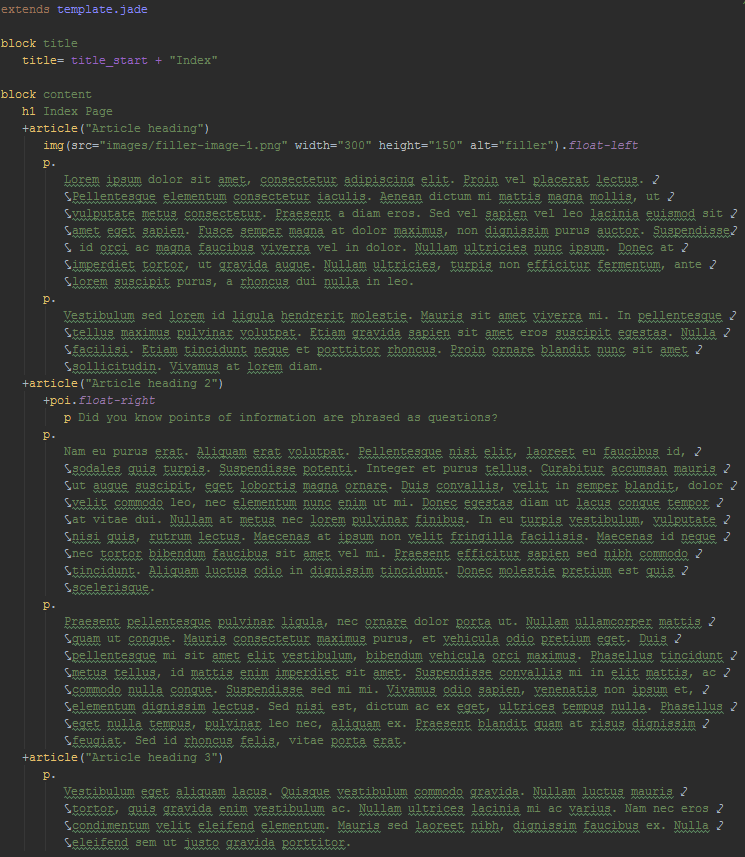
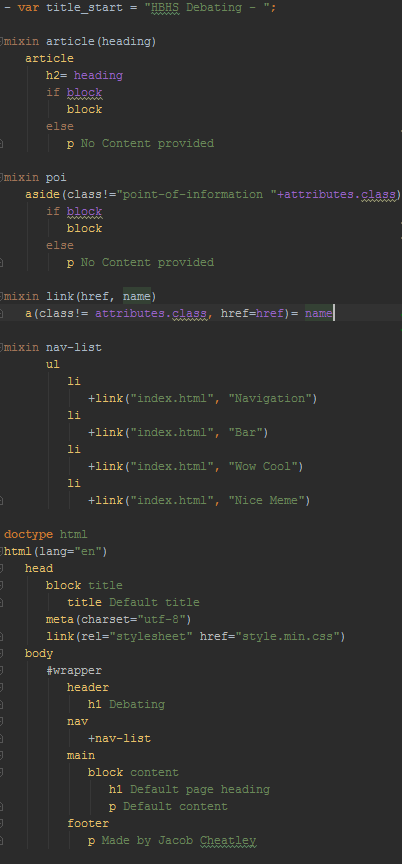
It is a fairly standard website using basic HTML and CSS. However, during my time creating it and searching the internet, I was drawn towards doing something more interesting than what I had done before. As a result, I decided to recreate my website using JADE and SCSS. Although this means I will not be able to edit my website at school (because setting up WebStorm to work properly with these files is a rather complicated process which would require admin rights to set up at school), it DOES mean I have access to much more powerful languages for creating my website, and I’ll learn a lot more. In addition, it means I can focus my time at school on writing content, taking and editing photographs, creating images, getting peer feedback, and adding to my evidence document.

The final result of this is similar to the one above, but it has a LOT more under the hood:



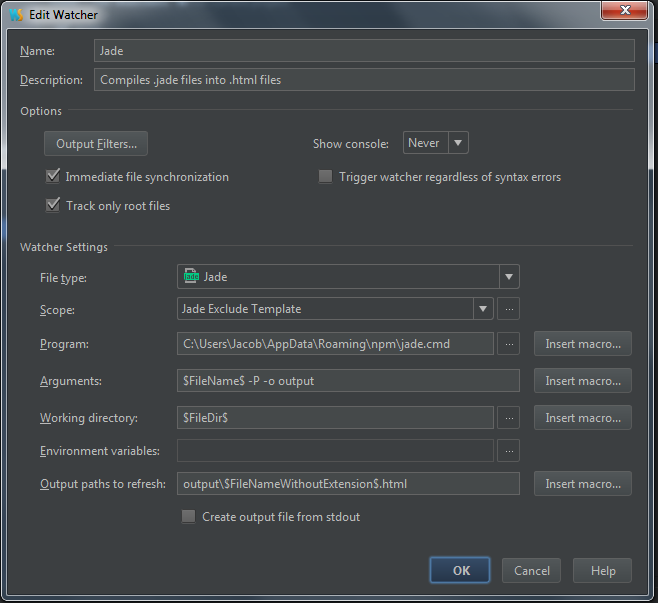
How this was achieved:

### Jade to HTML:

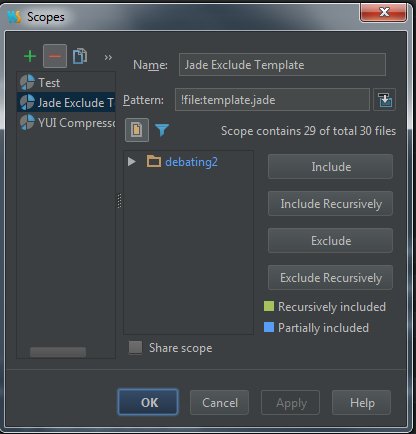


These are two Jade files, a template, and index which extends this template. Jade is an HTML Templating engine made in Javascript. The techniques that these two files use are detailed in the Jade techniques section.

Any Jade files which are not the template are watched by a file watcher as they update and are transpiled (compiled to another language, not an .exe) to html with a CLI command. This is the watcher dialog for Jade:

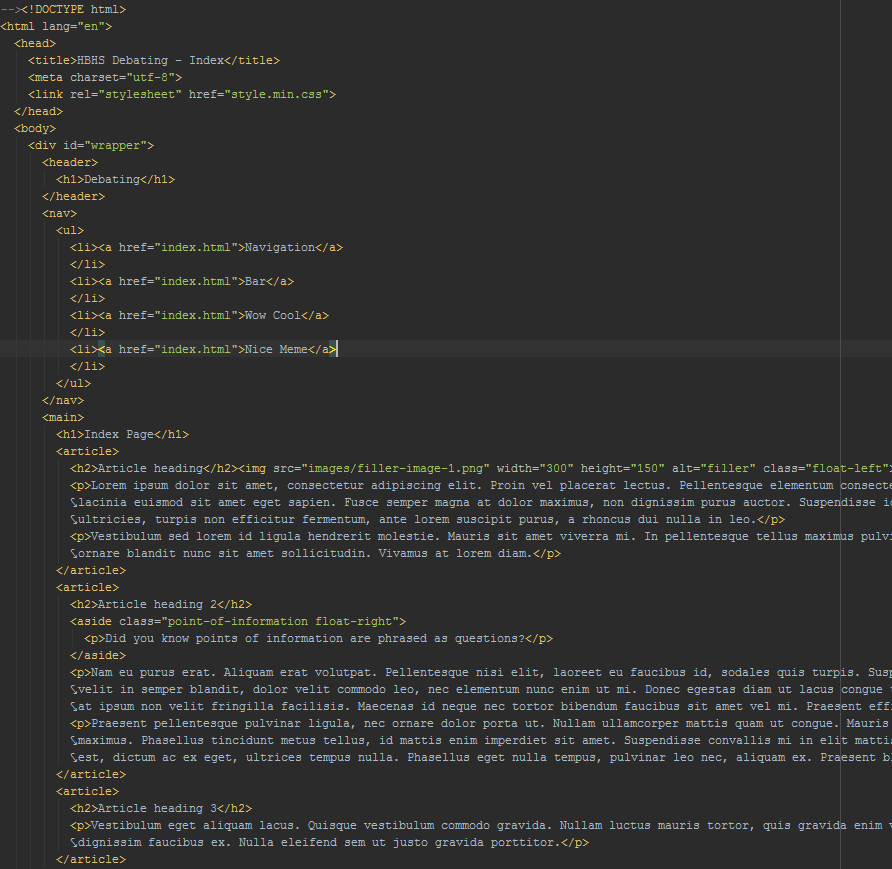


* File type: Jade files - it will only watch .jade
* Scope: ‘Jade Exclude Template’ – this is a custom scope I created to look for files which aren’t template.jade

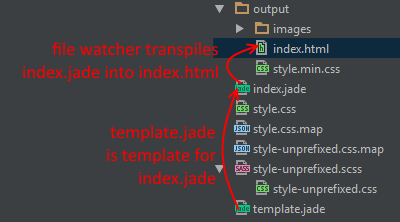
 (The key thing here is the pattern: !file:template.jade, which excludes the template)

* Program: jade.cmd, the command line where the arguments are run
* Arguments: This is where the work is done. This takes the file which the file watcher detects as updating and makes it into a ‘pretty’ (from –P flag) HTML file, outputting it in the directory output.

The resultant HTML file looks like this (the bottom and right are cut off for convenience because it is quite long, but you can see the end result more or less):



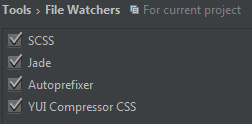
So effectively, all I do is write Jade files and the WebStorm file watcher I set up deals with the rest, putting completed HTML in the output directory. Sweet.



### SCSS to Prefixed CSS to Minfied CSS:

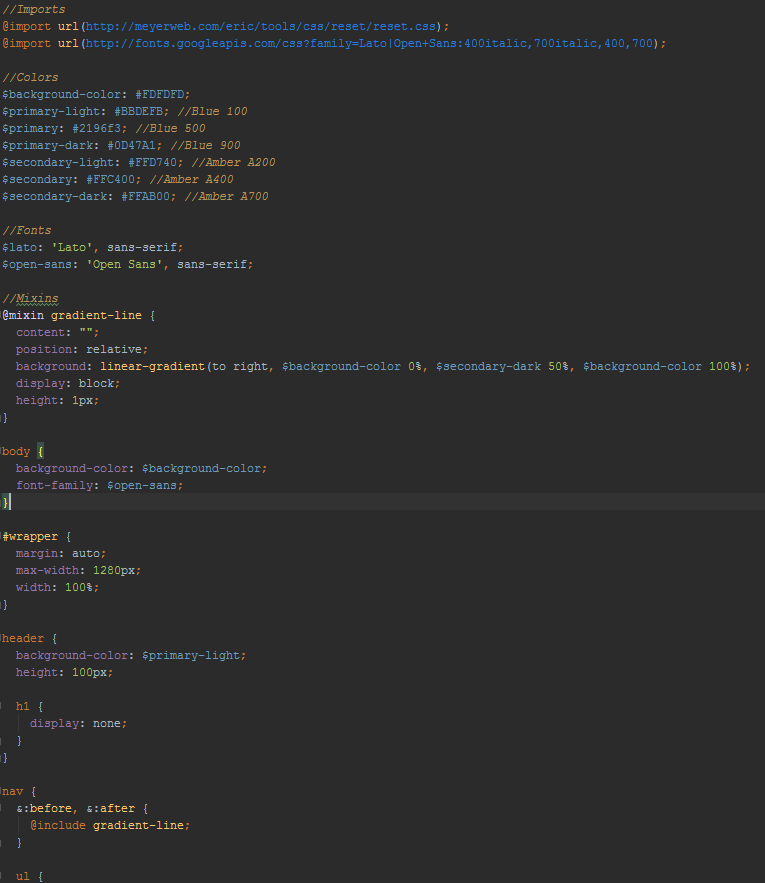
While my Jade -> HTML system only uses one file watcher, my SCSS -> (eventual) minified CSS system uses three.

I won’t bother explaining all of it in as much detail as the Jade -> HTML system.



Each of these file watchers has a different job. You’ve seen the Jade one.

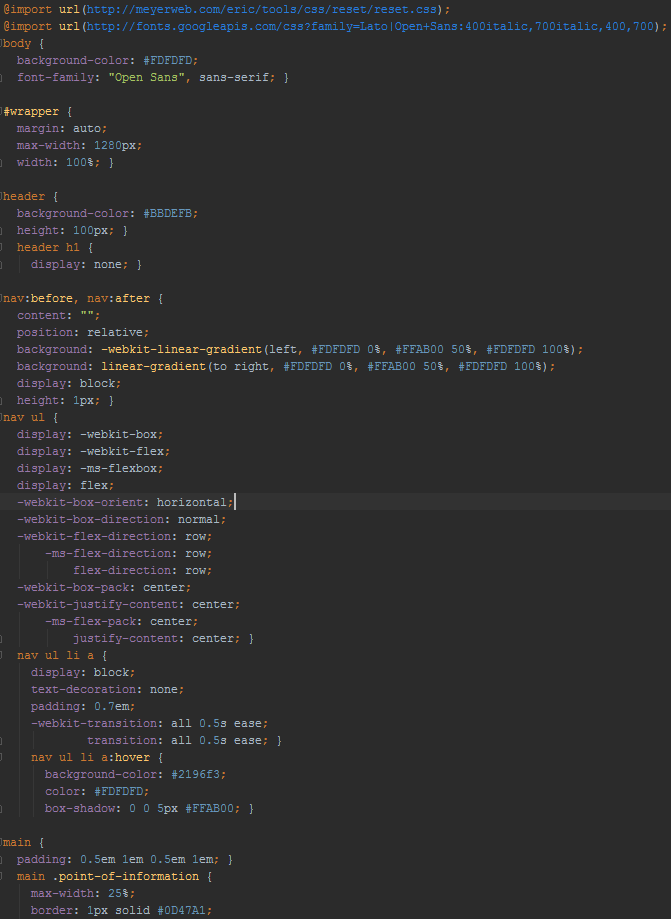
The SCSS one takes an SCSS file (shown is just a small part of it):



And converts it to a plain old CSS file:



But wait, we aren’t done quite yet! The Autoprefixer file watcher takes that CSS file and adds vendor prefixes to it to maximise browser compatibility:

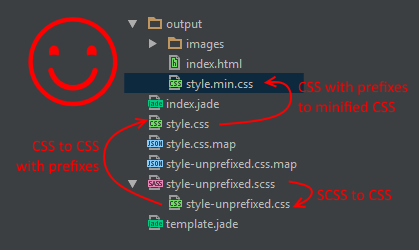


THEN, the YUI Compressor CSS takes that CSS file, and puts it all onto one line with as little spacing as possible for minimum file sizes:



Look at all of those word wraps.

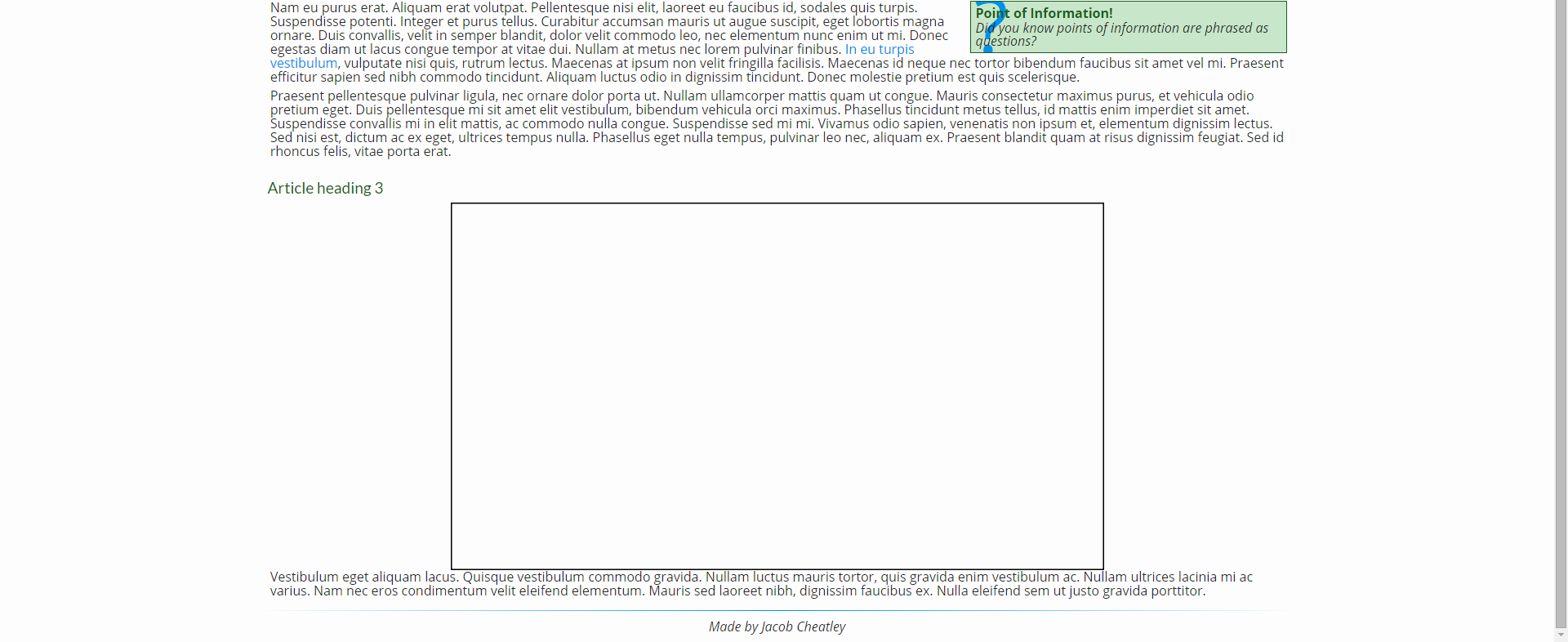
So, effectively I just write SCSS and out comes minified CSS complete with vendor prefixes, in the output directory.

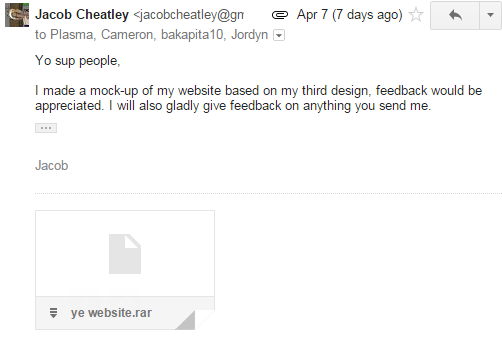


## Feedback and Iteration, Part I:

### Unguided Feedback (without asking specific questions):

I made quite a few modifications to my website and sent it off to some of the people I initially got feedback from:





Main feedback from people was:

Colour scheme is generally good (with the exception of the hover on the navs looking odd).

Font choices are good.

Header is boring (no image yet).

Nav text is purple (oops, easy fix)

A little more margins around content.

More margin around images.

### At School:

I showed my website to some people. Cameron said that stuff that doesn’t have a clickable action should not have a hover effect. I completely agree. This is because the hover should imply further action, or perform action itself. I will remove the box shadow around objects when hovered.

Briefly checking with the html validator I realise that I need to slightly change the argument for my jade output so as not to transpile comments.

Ezra:

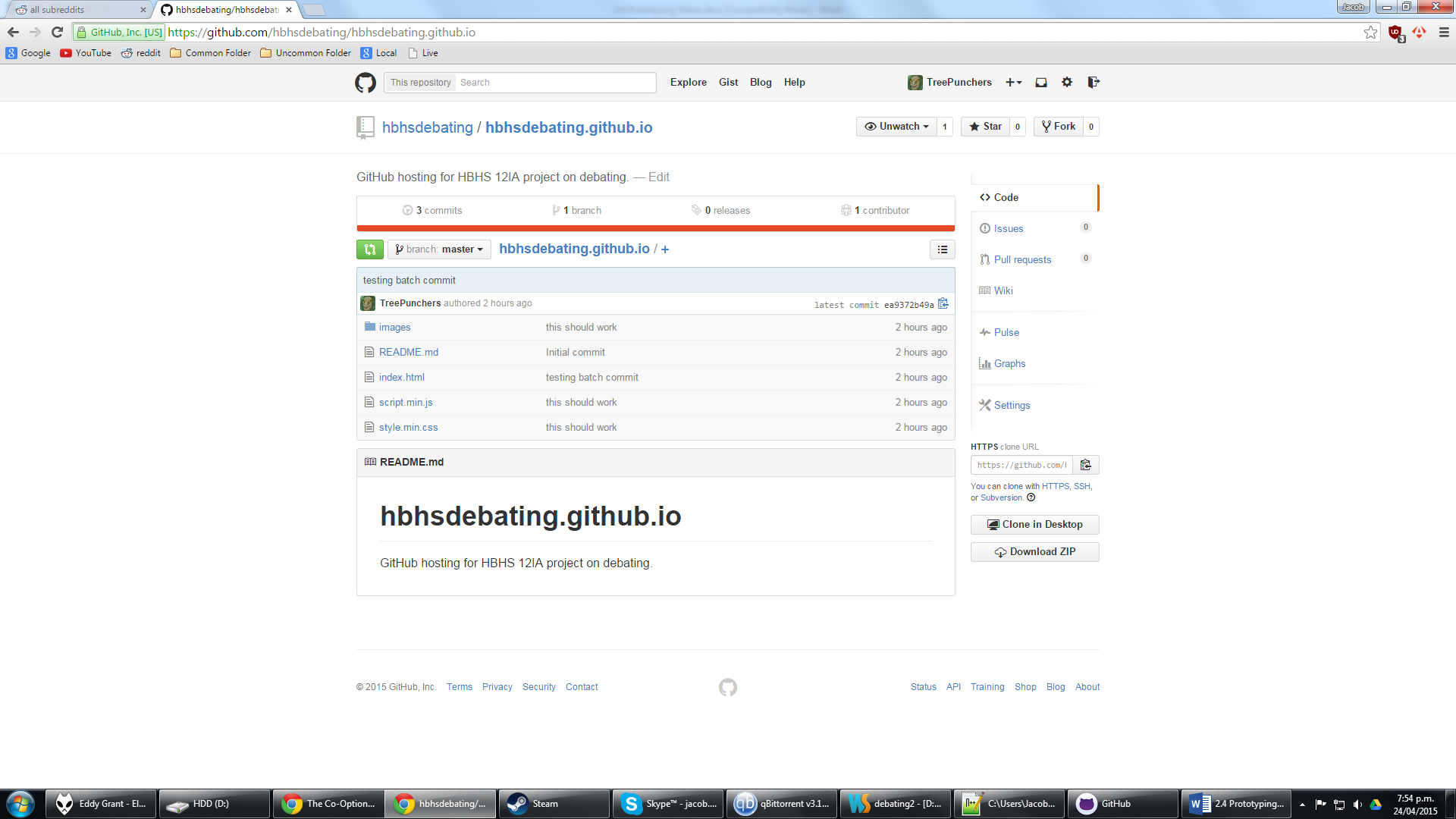
* Generally like the website. Mobile mode is nice and well functional.
* Margin on POI.
* Light green is bleh colour, and the dark green on nav is odd as well. (I certainly will change this up).

## Hosting the Website:

I set up a GitHub repository on my own account to store all of my files - <https://github.com/TreePunchers/12IA-Debating-Website>:

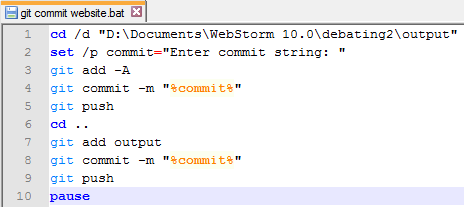


This also has a submodule which is on an organisation I have created called hbhsdebating - <https://github.com/hbhsdebating/hbhsdebating.github.io>:



This is for the hosting of the website (in addition to having all the other great features that hosting on GitHub gives.

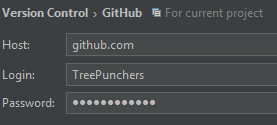
The website can be found live at <http://hbhsdebating.github.io/> ☺ Every time that I commit and push changes to my GitHub this will update. There is one issue however, as I can’t commit the main repository without the submodule also being fully committed (and probably even pushed I think). Which is inconvenient, as I can’t actually commit or push from anything other than issuing 4 separate commands in a CLI. However, I still wanted to use a system structured like this so I made a batch file to automagically commit and push all changes in the submodule and the main repository.



Which works a treat.

Oops, it turns out I didn’t even need to do that. WebStorm can commit and push to GitHub with a bit of setting up. Awesome.

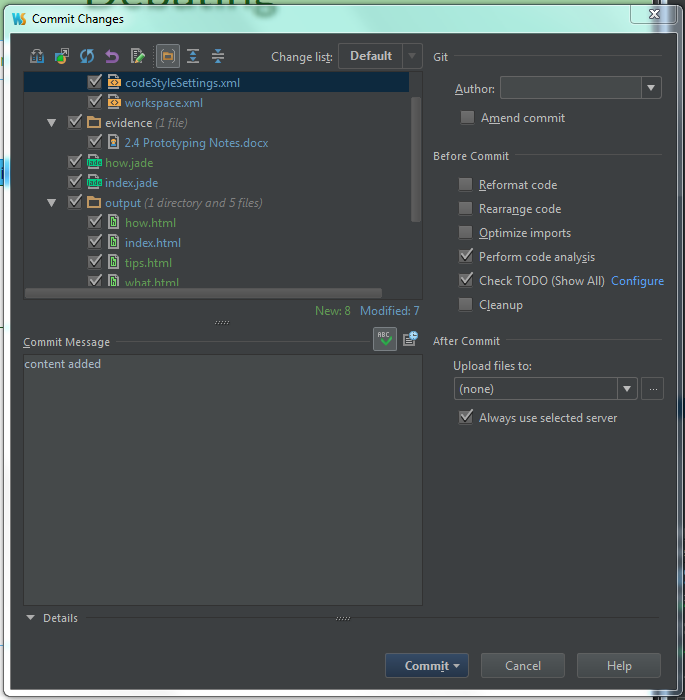
All that requires is linking your GitHub account to WebStorm:



And then when you want to commit/push just use the little button in the top right toolbar:



(shortcuts are pull, push, show history and revert)



## Content:

I made a content.