**The Anatomy of a Website**

Websites are made up of several major components:

* **Page –** A single document viewable by the user containing information
* **Styles –** Documents or code that describe how the page should be rendered (shown)
* **Actions** – Implemented as code, not viewable in the browser, that performs some action

Actions can be of several types:

* **In-page** – the most commonly-known are hyperlinks, links to other webpages
* **Client-side** – like in-page actions, these perform some action (like hover actions, zoom etc.) in the browser
* **Server-side** – these are actions that make a call to some central server(s), such as for serving ads

Languages – there are several languages used for creating websites

* **HTML** – HyperText Markup Language. A ‘tagging’ system with defined components that are assembled into the structures that make up the page.
* **CSS –** Cascading Style Sheet. A special type of page used mostly for styling HTML components.
* **Javascript –** A client-side language for performing **actions.** There are several variants like JQuery, React.js
* **PHP –** A language for performing both client- and server-side actions
* **Other languages -** ASP.NET, Python, Ruby – for creating fully-fledged web applications

Frameworks – these are off-the-shelf website creation frameworks and come in various levels of complexity.

* **Wordpress** – A WYSIWYG (what-you-see-is-what-you-get) framework used by inexperienced creators of websites to put together pages and actions, and style cosmetics.
* **Ruby on Rails** – A framework based on the Ruby programming language to create web applications.
* **Django** – A framework based on the Python programming language to create web applications.

In the early days of the Internet, websites used to be a few pages in number and directly linked (using hyperlinks) into simple hierarchical structures. However today, there is much more that goes on when you use a web application. Some of the things that happen when you visit a news site, for example, include:

* Your IP address, browser type, approximate location are logged and sent to a web server
* Ads that target you or your demographic are shown in special areas on the screen
* Custom Javascript will run to launch pop-ups, for example for GDPR, or to cross-sell a service
* If you reached the page from an affiliated website a record of your visit is logged and sent to the referrer
* Your mouse movements and pages you click on may be tracked
* The browser may ask you to allow notifications, or access to your linked services

Companies who host websites will keep track of many metrics about their users in real-time. These include:

* Number of users actively browsing the website
* Contents of users baskets
* Length of time spent browsing products
* Links clicked and navigation path through the website in general
* Interest in certain areas of the page (achieved by tracking mouse movements)

Rendering the text is actually the easiest part of the whole process for displaying a website. On the next page we will look at exactly what the source code of a basic web page might look like.

**Activity:**

* Go to BBC News (<https://www.bbc.co.uk/news>), right-click and View Source. Familiarise yourself with what you find.

**Basic Web Page Structure**

Pages are composed using HTML. HTML pages have two main areas, the **head** and the **body**. We use a tagging system to mark these areas. Inside these areas, we use more tags to define specific parts of the page.

In its most basic form, a webpage could look like this:

<html>

<head>

<title>This is Cydney's webpage!</title>

</head>

<body>

<h1>This is the title of my webpage</h1>

<h2>Underneath the title, we have a slightly longer description.</h2>

<p> Now the main text begins. We mark out each paragraph using the p tag. Whitespace (empty space) in webpages doesn't matter - the browser ignores repeating spaces, which is convenient when designing them. It's not very convenient when we want to add our own &nbsp w &nbsp&nbsp e &nbsp&nbsp i &nbsp&nbsp r &nbsp&nbsp d elements to the page, though. We always end a tagged section with a matching tag preceded by a forward-slash.</p>

<p>Now let's create a hyperlink. <a href="https://www.google.com">Clicking this link</a> takes you to Google.</p>

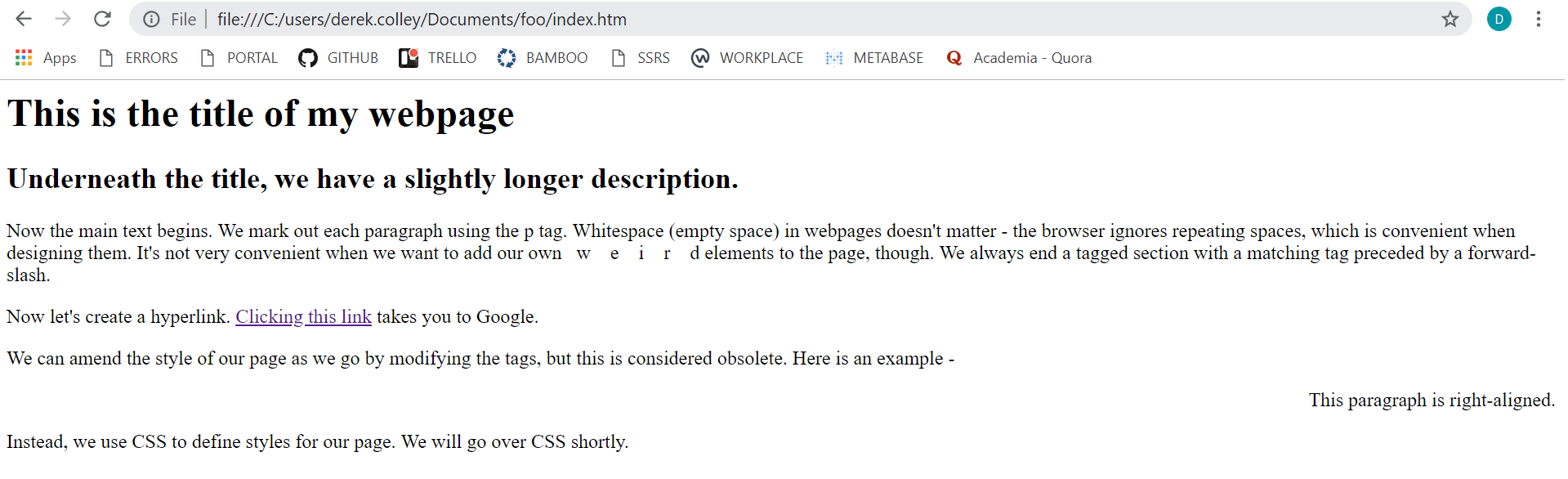
<p>We can amend the style of our page as we go by modifying the tags, but this is considered obsolete. Here is an example - </p>

<p align="right">This paragraph is right-aligned.</p>

<p>Instead, we use CSS to define styles for our page. We will go over CSS shortly.</p>

</body>

</head>



**Activity:**

* Copy and paste the code above into a new Notepad document. Save the document as index.htm. Now navigate to and open the document in File Explorer, which should open it in a new browser window.

**Styling a Webpage**

There are two main kinds of styling – **inline** or **internal** styling, and **external** styling using **stylesheets**. These are both accomplished using the same language, CSS. Inline styling belongs in the HTML tags or head tag of the HTML document and looks like this:   
<p style=”font-family: Courier New;”>

Using a stylesheet, however, we can define styles for many different tag types, ids and classes, which gives us much more flexibility when creating a website. A stylesheet is a **file ending in .css** containing our style definitions. We reference an external stylesheet in the <**head>** tag of the website, like this.

<head>

<title>This is Cydney's webpage!</title>

<link rel="stylesheet" href="styles.css">

</head>

Using this example, we can define the following stylesheet.

body { background-color: green;

padding: 25px; }

h1 { font-family: “Courier New”;

font-size: 2.0em; }

h2 { font-family: “Calibri”;

font-color: yellow;

font-size: 1.2em; }

p { font-size: 0.8em;

text-align: center; }

**Activity:**

* Amend your webpage to include the <link> tag as shown above.
* Copy and paste the CSS stylesheet code into a new Notepad document and save it as styles.css in the same directory as your webpage.

**More HTML and CSS Elements**

There are hundreds of different HTML tags available but in practice, most websites only need a small handful.

<h*n*>Headings h1 to h6, different types of headings</h*n*>

<p>Paragraph</p>

<a href=”url”>Hyperlink</a>

<img src=”url”>Image. URL links to a picture file or online URL and doesn’t need a closing tag.

<button>Click this button.</button> Actions can be set for when the button is pressed.

Tables are more complex, and have nested elements.

<table>

<th> Customer Name </th>

<th> Sale Amount </th>

<tr>

<td>Bob</td>

<td>£15.00</td>

<td>Jim</td>

<td>£8.00</td>

</tr>

</table>

<div>Start of a division. We use div to separate out elements of our site, normally for styling.</div>

We can assign an **id** or a **class** to various HTML tags then in the CSS, define a style for the id or class. We use the ‘class’ keyword in the tag, and reference it using the tag type, plus a dot(.), then the class name in the CSS. Here’s an example.

**HTML:**

<div class=”BigBoldText”>

<p>This text is large and bold, and is designed to catch the eye.</p>

</div>

<div class=”NormalText”>

<p>Whereas this text is normal, contains the detail. Notice how each div has its own class.</p>

</div>

**CSS:**

div.BigBoldText {

font-size: 2.5em;

font-weight: bold;

}

div.NormalText {

font-size: 1.0em;

}

**Activity:** Modify your webpage to include a stylesheet. Then, amend the stylesheet to do the following.   
Use <https://www.w3schools.com/html/> and <https://www.w3schools.com/css/> for help.

* Centre all text (hint: use text-align and the American spelling, center)
* Modify the letter spacing for all <p> blocks of text to 3 pixels (hint: use letter-spacing)
* Change the background image to an image of your choice (hint: use background-image and find a suitable URL)
* Create a division (div) and put some text in it. Make this text twice the size of all other text on the page.

**Web Hosting**

There are several prerequisites before you can get your website online.

* **Domain Name** – you need to own the domain name for your website. This means you are able to amend the records associated with that domain name. There are several types of records including A records (which server the domain name should point to) and MX (which mail server any email addressed to that domain should point to).   
  A domain name is registered using a **registrar**. These are large organisations who are charged with administering TLDs (top-level domains).
* **Web hosting** – You need access to a server to host your files. A server is simply a computer system that can respond to incoming requests for web pages from users and serve them to the user. Web servers can be small and simple, or large and complicated. A popular model is to use an external third-party to host your site.
* **Web content** – your content must be written and ready to upload. HTML/CSS and other client-side content can normally be tested locally (this means on your computer). Server-side content may be more complicated to test. For larger organisations, a whole CI/CD (continuous integration/continuous development) pipeline may be configured to continually push out website updates.
* **Server-side services** – For more complicated websites you might need a **database**, or to interface with other websites or systems. For example, if you created a website which needed users to register an account to e.g. shop, you would need to store their information somewhere. Likewise, you might keep your inventory in a database so that you can simply upload new stock details directly to the database without having to keep pushing out website updates.

**Demo –** Uploading a website to web hosting

**Demo –** Source control systems