



School of Computer Science

IS5103 Web Technologies: Introduction

Dr Ruth Letham





Module Information

Followed by Q&A





Key Information

- You are assumed to be familiar with the whole student handbook
 - Programme & Module information
 - Assessment, Marking, Lateness penalties
 - Key Policies & Procedures
- Read the key points from student handbook

The screenshot shows the homepage of the 'CS Student Handbook' website. At the top, there is a search bar with the placeholder 'Enter search keywords' and a 'Search Student Handbook' button. Below the search bar, the word 'Welcome' is prominently displayed in bold black text. A brief introduction follows, mentioning the School of Computer Science at St Andrews and its relevance to students in the 2019-20 session. It encourages reading the whole handbook and provides links to induction materials for various classes (First Year, Second Year, Honours and MSci, MSc). Further down, it notes that the handbook avoids duplication with other University publications and lists links to the University Student Handbook, Course Catalogue, Rules and Regulations, and Senate Regulations. A section titled 'Wellbeing, Advice, and Support for Students' is also visible.

<https://info.cs.st-andrews.ac.uk/student-handbook/>





Key information (cont.)

- Read the [Good Academic Practice policy](#)
- Check that coursework submitted to MMS has been received successfully, and that it's the right piece of coursework
- Coursework submitted after deadline is subject to automatic penalty
- Any special circumstances must be documented immediately through the self-certification system
 - and followed up with coordinator if you are seeking any allowance
- You must be available for the entire exam period
- Familiarise yourself with the [School](#) and [University](#) health & safety guidance





Lecture Topics

- Introduction to web standards
- Evolution of mark-up Languages
- Standards and Cascading Style Sheets (CSS)
- Managing Web projects
- Interactivity
- Web Accessibility
- Responsive and sustainable

Learning Outcomes

On successful completion of this module, the student should:

- Have gained knowledge and understanding of modern web standards and the techniques and tools for implementing, testing and evaluating web sites.
- Have gained knowledge and understanding of Web Accessibility and the Web Content Accessibility Guidelines.
- Be able to implement web sites using modern frameworks and a variety of resources.

Syllabus

- Web standards.
- Semantic information mark-up.
- Presentation logic using Cascading Style Sheets.
- Web project management.
- Navigation and interaction.
- Accessibility and usability basics.
- Technologies for developing responsive web pages.

<https://info.cs.st-andrews.ac.uk/student-handbook/modules/IS5103.html>





Module Assessment

- One essay (30%)
 - Research a topic & support/refute a position
 - Due ~week 7
- One practical assignment (30%)
 - Research a topic & develop a website
 - Due ~week 10
- 3 hour online exam (40%)
 - During December exam diet
- 20-point scale for assignment marks
 - A mark is given to an individual piece of coursework
 - Each assignment is allocated a weight





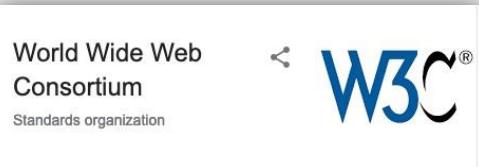
Module Delivery

- Lectures
 - To delivery core content
 - Will be recorded ‘as live’ for revision purposes
 - › On a ‘best effort’ basis
- Preparation for lecture distributed in advance
 - On studres <https://studres.cs.st-andrews.ac.uk/IS5103/>
- Exercises
 - To develop practical skills
 - › HTML, CSS, Accessibility tools, frameworks, etc





Online Resources



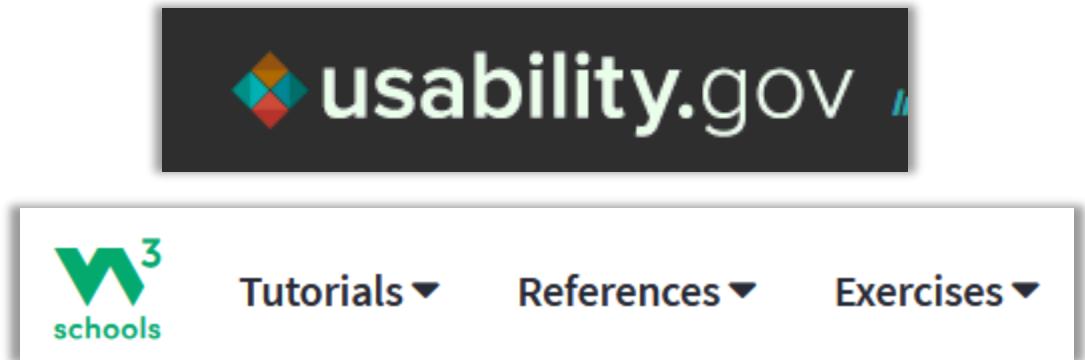
readwrite



Sustainable Web Design

[View strategies](#) [Calculating Digital Emissions](#)

Web technology has the potential to bring huge benefits to society and the environment, but only if we use it wisely...





9

Any Questions?





What is the web?





A trip back in time...

1989 Tim Berners-Lee Invents the World Wide Web

- Frustrated by the need to log into different computers to access different information, often stored in different formats
- Invented WWW as a communications tool to allow anyone, anywhere to share information
 - Whatever their hardware, software, network infrastructure, native language, culture, geographical location, or physical or mental ability





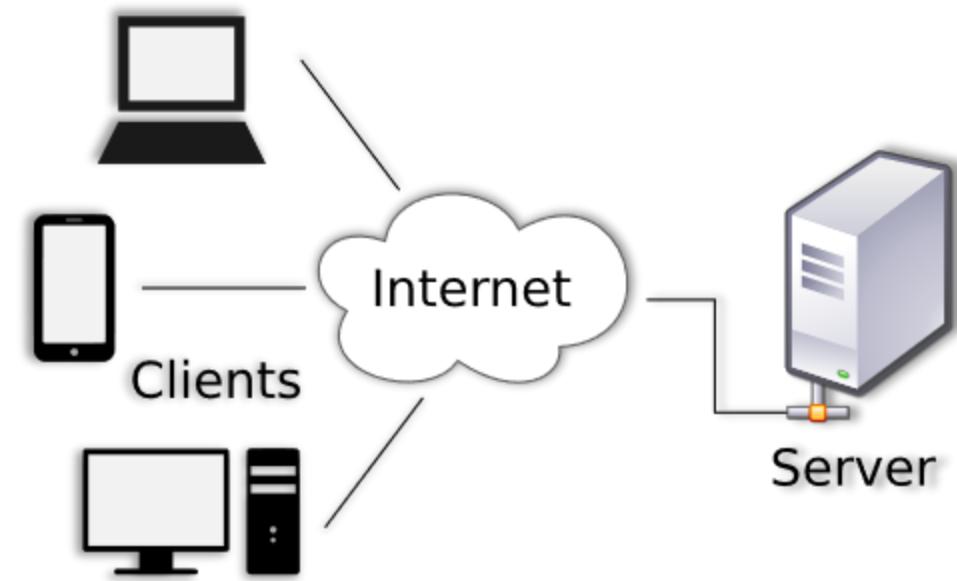
WWW Technologies

- Combination of different technologies
 - Transmission Control Protocol
 - Internet Protocol
 - Domain Name Servers
 - Markup
 - Hypertext
- HyperText Transfer Protocol (HTTP) for communication
- HTML, XML & XHTML mark-up for content
- Cascading Style sheets (CSS) for visual design
- Document Object Model (DOM) underlying representation of document
 - Often used with JavaScript as a programming interface



Web communication

- Servers hold information
 - E.g. web pages
- Client *requests* information from server via the internet
 - URL
 - HTTP header
- Server *responds* to client
 - With a status code
 - And if all goes well, some content
- Client processes response
 - E.g. displays web page in a browser



[Gnome-fs-client.svg](#): David Vignoni [Gnome-fs-server.svg](#): David Vignoni derivative work: Calimo, [LGPL](#), via Wikimedia Commons





Status 404



Welcome to 404 error page!

Welcome to this customized error page. You've reached this page because you've clicked on a link that does not exist. This is probably our fault... but instead of showing you the basic '404 Error' page that is confusing and doesn't really explain anything, we've created this page to explain what went wrong.

You can either (a) click on the 'back' button in your browser and try to navigate through our site in a different direction, or (b) click on the following link to go to homepage.

[Back to homepage »](#)





How do you request a web page?

- Send an HTTP request to a host
 - Request line includes method, path component of the URL, HTTP version number
 - Header tells the server about the message – so it can interpret it properly
 - Body contains content (if there is any)
- Web browsers build your HTTP request for you
 - Just give them a URL
- URL (Uniform Resource Locator) is a special type of URI (Uniform Resource Identifier)
 - Standard way to refer to a location
 - Includes: protocol, domain name, path, file name (optional)
 - Example:
<https://studres.cs.st-andrews.ac.uk/IS5103/Lectures/Week-1-Introduction/1-Introduction.pdf>





HTTP

- Every HTTP request includes all information necessary to fulfill that request
 - i.e. they are *stateless*
- HTTP data is **not** encrypted
- HTTPS data is encrypted (the "S" stands for *secure*)
 - Uses an SSL certificate for encrypted connection
 - Used for communication of any sensitive data
 - › e.g. card payments, login credentials, etc.





HTML

- Tags used to ‘mark-up’ content
 - Covered in depth in Week 2

<h1>A simple mark up language</h1>

<p>HTML provided a document outline with a basic tag syntax used to describe content and create hyperlinks.</p>





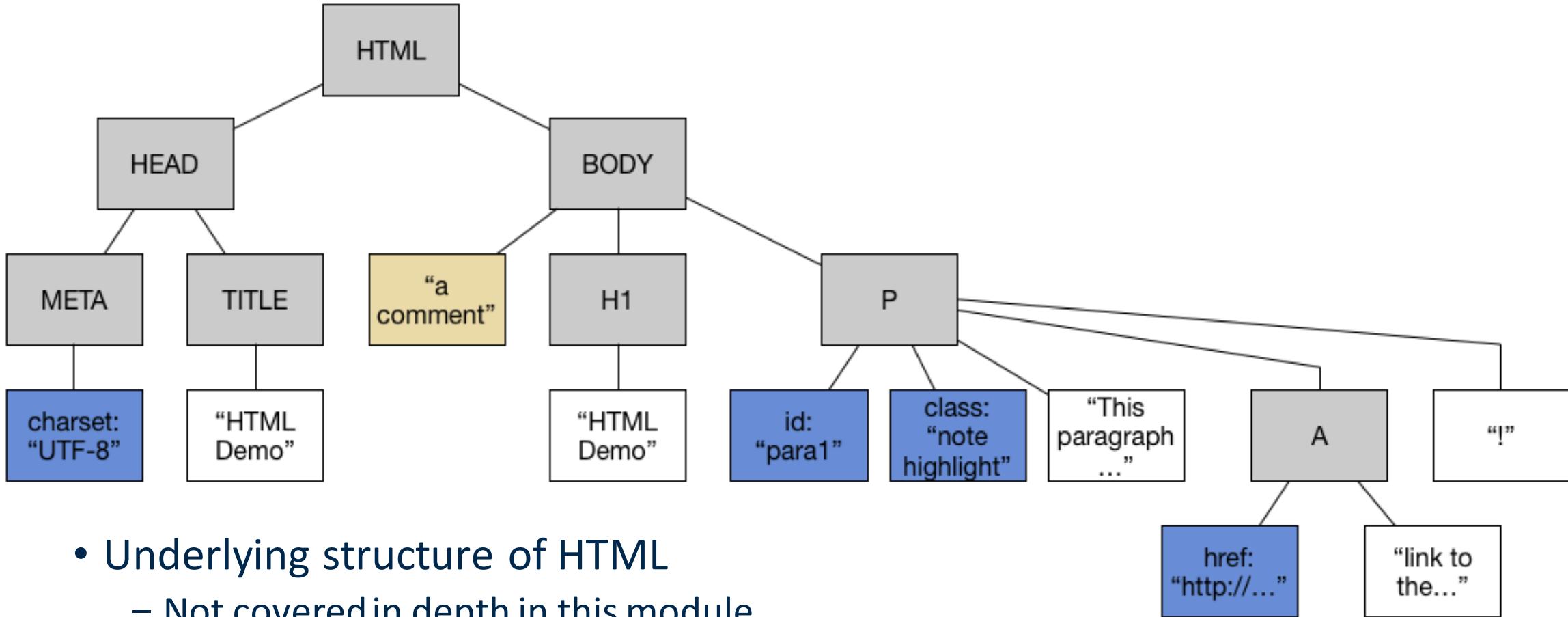
CSS

- Rules used to describe visual display of HTML elements
 - Covered in depth in Weeks 3 & 4

```
body {  
    color: #325050;  
    background: #fff;  
    font-family: 'Libre Baskerville', sans-serif;  
    font-size: 70%;  
}
```



DOM



- Underlying structure of HTML
 - Not covered in depth in this module





What are web standards?





Web Standards

- Define syntax and semantics of web protocols & languages
- Interdependent specifications
- Open and publicly documented specifications
- No license or copyright
- Responsive to users evolving needs
- Common ground for web development





Web Standards Organisations

- Technologies established by standards development organisations
- Internet Engineering Task Force (IETF)
 - Underlying protocols
- World Wide Web Consortium (W3C)
 - General web standards
- Web Hypertext Application Technology Working Group (WHATWG)
 - HTML specification
- European Computer Manufacturers Association Technical Committee 39 (ECMA TC39)
 - JavaScript standards





Web Standards Process

W3C Standardisation Process	
Working Draft	Interested parties for general review
Last Call Working Draft	Technical requirements & dependencies
Candidate Recommendation	Published to test implementation
Proposed Recommendation	Extensive review & Testing
Proposed Edited Recommendation	Important changes review
Recommendation	Endorsed by W3C's Director

<https://www.w3.org/2021/Process-20211102/#rec-track>





Why do we need web standards?

Consistency, compatibility & quality



Compatibility: The First Browser War

- Late 1990s
 - Rapid addition of new features
 - Proprietary languages & tags
 - ‘best viewed in ...’
 - ‘Viewable With Any Browser’ campaign



- Without agreement on how to describe and send information how can we have universally accessible communication?





Potential Problems

- Work on a certain browser
- Print badly
- Unreadable on small devices
- Inaccessible to assistive technology
- Complex to redesign
- Look out of date
- Websites are Business Critical
 - ...so problems could have major consequences!





Mitigating problems

- Separation of concerns

- Content

- › Information you want to convey
 - › Meaning of data
 - › Meaningful mark-up

- Presentation

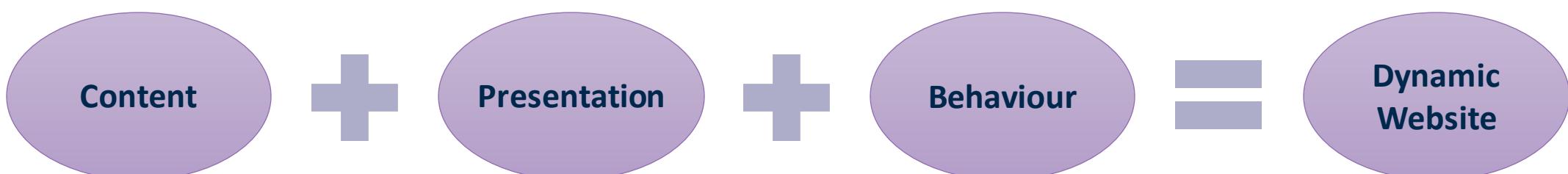
- › Corporate identity & style

- Behaviour

- › JavaScript

- General benefits

- Reduced workload
 - Reduced risk of errors
 - Broader availability



<https://www.thoughtco.com/three-layers-of-web-design-3468761> Accessed August 2021





Separation of concerns: Design & Presentation

The image shows two screenshots of the CSS Zen Garden website. The left screenshot displays a dark-themed design with a green background image, featuring a large white circular logo and the text 'CSS ZEN GARDEN' and 'The Beauty of CSS Design'. Below this is a text block about CSS-based design and links to download example files. The right screenshot shows a light-themed design with a blue background, featuring a large white 'VIEW ALL DESIGNS' button. Both screenshots include a sidebar on the right listing various design styles like 'MID CENTURY MODERN' and 'STEEL'.

A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example [HTML FILE](#) and [CSS FILE](#)

THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WASP, and the major browser creators.

MID CENTURY MODERN
by Andrew Lohman

GARMENTS
by Dan Mall

STEEL
by Steffen Knoeller

APOTHECARY

VIEW ALL DESIGNS >

CSS ZEN GARDEN

The Beauty of CSS Design

Select a Design:

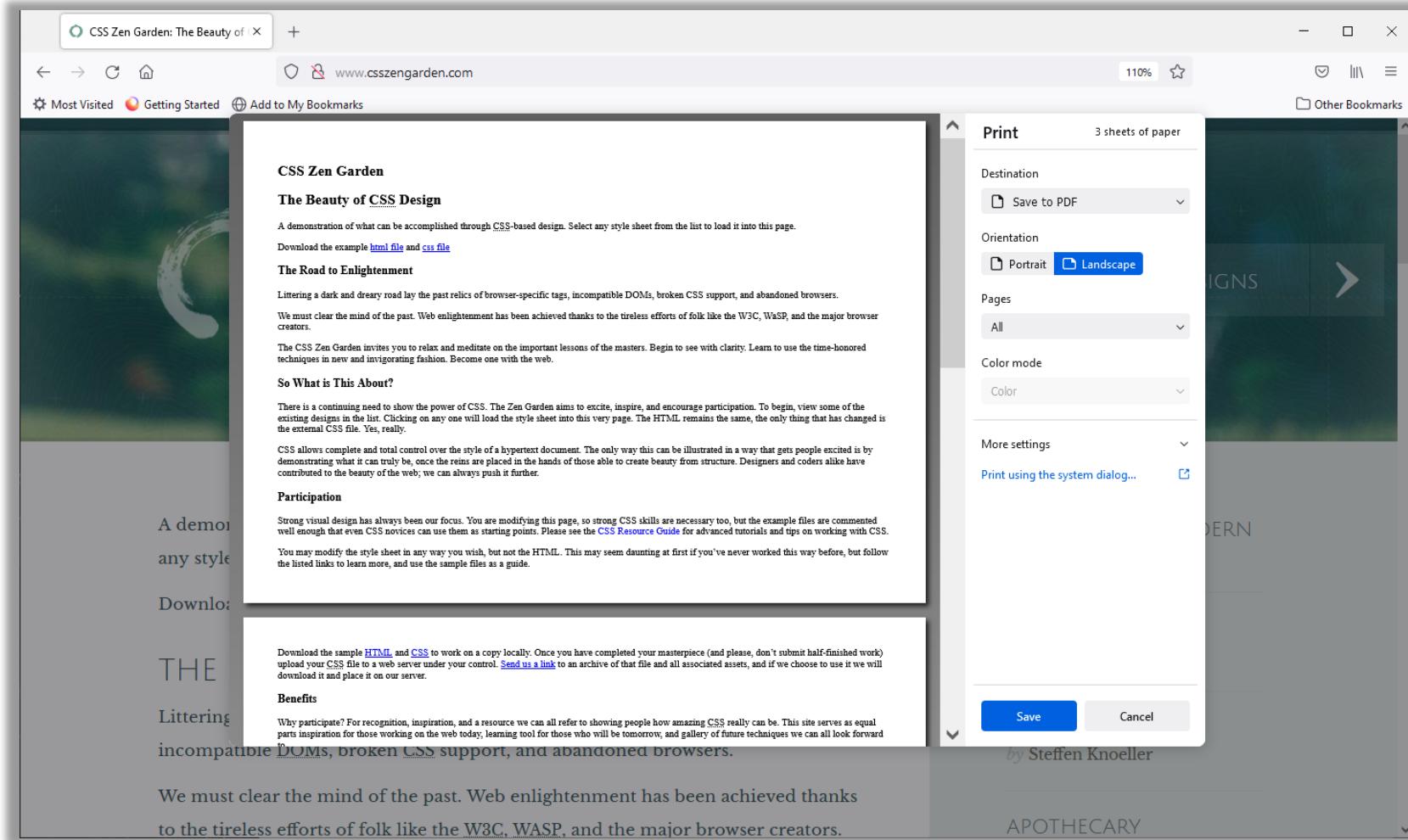
- Mid Century Modern by Andrew Lohman
- Garments by Dan Mall
- Steel by Steffen Knoeller
- Apothecary by Trent Walton
- Screen Filler by Elliot Jay Stocks
- Fountain Kiss by Jeremy Carlson
- A Robot Named Jimmy by meltmedia
- Verde Moderna by Dave Shea

```
51 <body id="css-zen-garden">
52 <div class="page-wrapper">
53
54   <section class="intro" id="zen-intro">
55     <header role="banner">
56       <h1>CSS Zen Garden</h1>
57       <h2>The Beauty of <abbr title="Cascading Style Sheets">CSS</abbr> Design</h2>
58     </header>
59
60     <div class="summary" id="zen-summary" role="article">
61       <p>A demonstration of what can be accomplished through <abbr title="Cascading Style
62       <p>Download the example <a href="/examples/index" title="This page's source HTML coc
63     </div>
64
```

© School of Computer Science



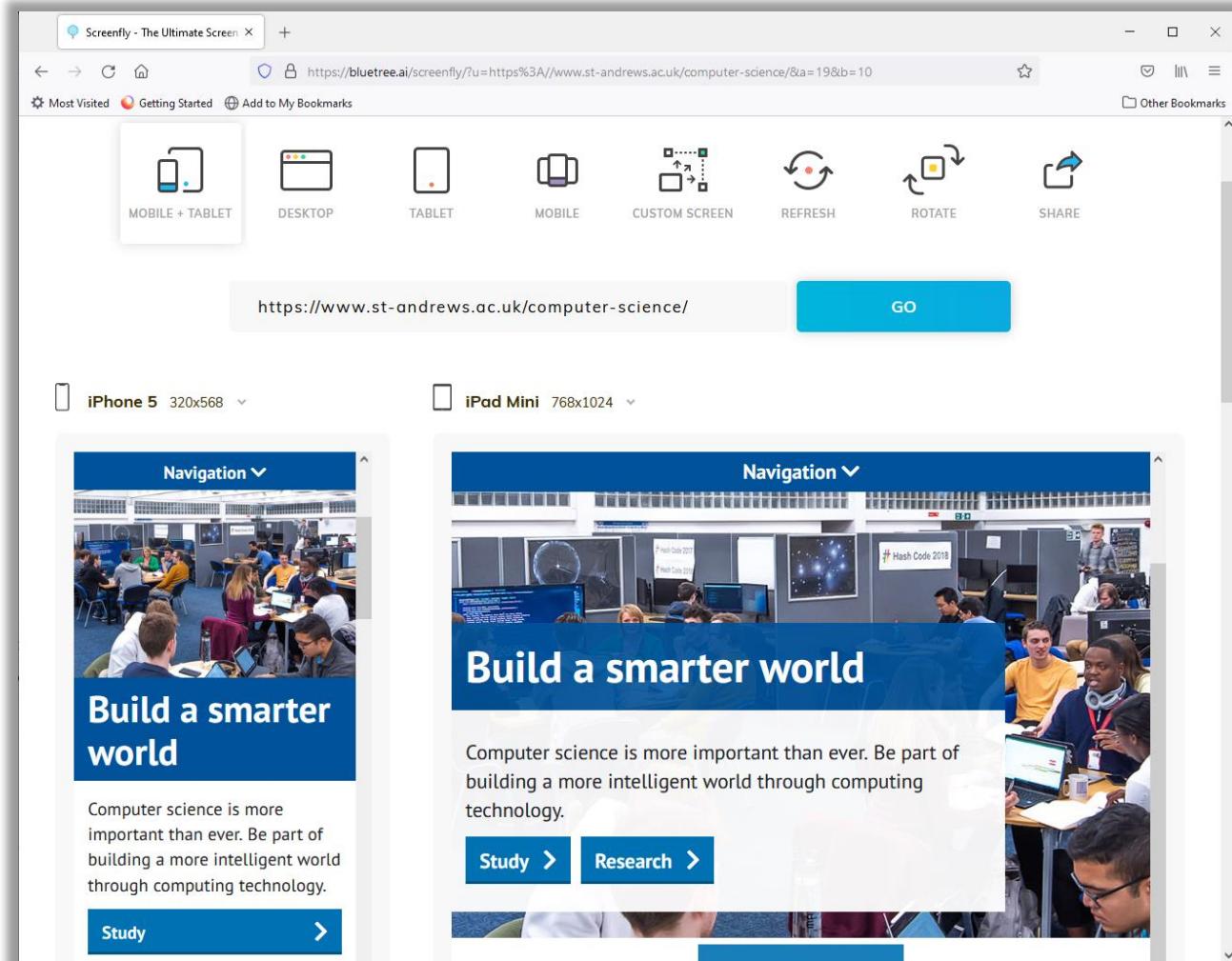
Separation of concerns: Preview & Print





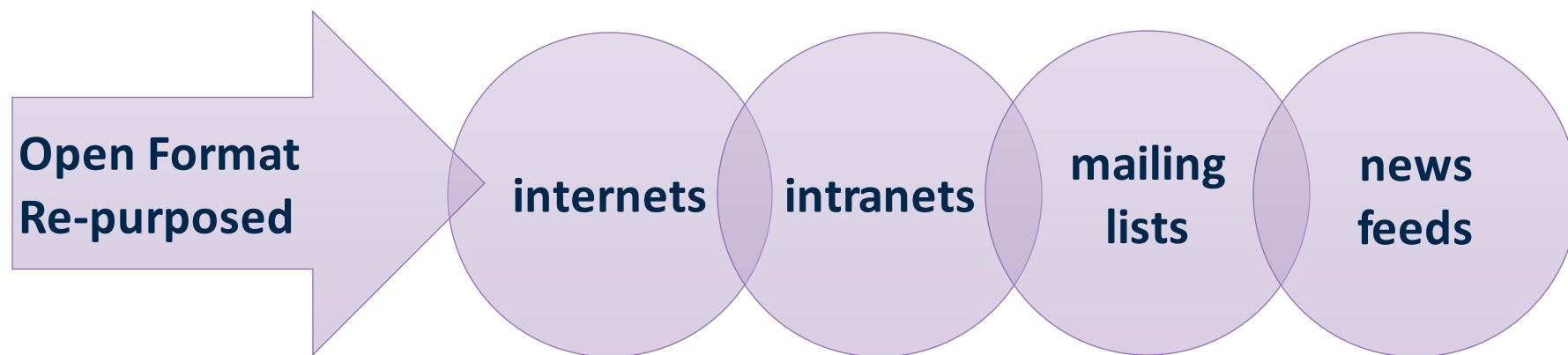
Separation of concerns: Devices

- Proliferation of mobile devices
- Test website using
 - in-built browser tools
 - or web sites like
<https://bluetree.ai/screenfly>
- Poor usability = lost business



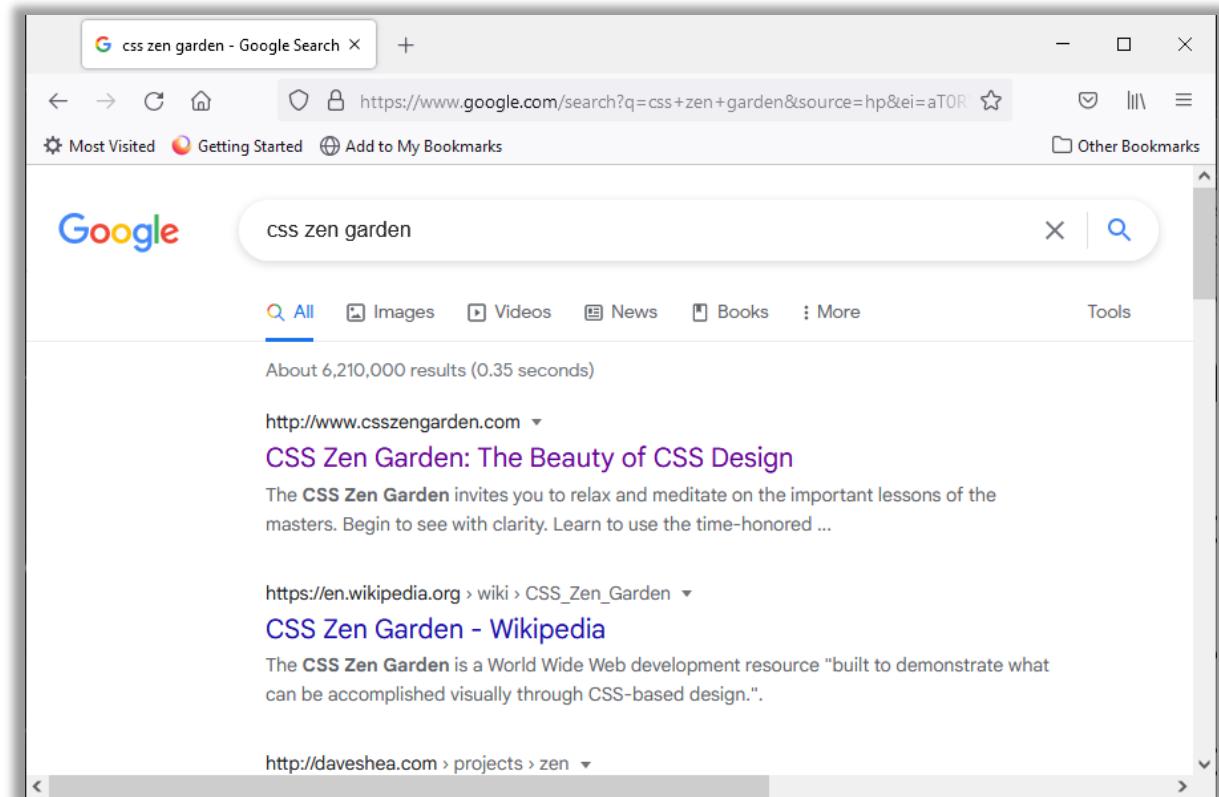
Meaning of data: Meaningful Mark-up

- Meaningful Structure of Content
- No embedded presentation information
- Neutral open format



Meaning of data: Search Engines

- Adherence to standards affects rank
 - Malformed mark-up = lower rank
 - Misuse of tags = lower rank





General Benefits

- Browser Independence
 - Old & New
- Compatibility with user agents & devices
 - Mobile Devices, Search engines, Screen Readers
- Standard output from proprietary website building software & content management systems





Validation

- We have standards, so we can
 - Assess standards compliance
 - Validate pages & sites
- Should all pages validate?

The W3C Markup Validation Service

Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI Validate by File Upload Validate by Direct Input

Validate by URI

Validate a document online:

Address:

More Options

Check

Showing results for http://www.csszengarden.com/

https://validator.w3.org/nu/?doc=http%3A%2Fwww.csszengarden.com

Most Visited Getting Started Add to My Bookmarks

http://www.csszengarden.com/

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 The `banner` role is unnecessary for element `header`.
From line 55, column 3; to line 55, column 24
`intro"><header role="banner">`
2. Warning The `complementary` role is unnecessary for element `aside`.
From line 112, column 2; to line 112, column 45
`</div><div class="sidebar" role="complementary">`
3. Warning The `navigation` role is unnecessary for element `nav`.
From line 117, column 5; to line 117, column 27
`</h3><nav role="navigation">`
4. Warning The `navigation` role is unnecessary for element `nav`.
From line 141, column 5; to line 141, column 27
`</h3><nav role="navigation">`





Should all pages validate?

- Improved Search listings
- Good publicity(?)
- Less expensive to develop & test
- Faster site
- More Visitors





Evolution & Future of the web





Evolution of the Web

- A service with clients on many platforms
- Create once, publish everywhere
- Legacy User Agents
- Geeks, Marketing and Bean-counters





Future of the Web

- Longevity of web-based information
- Interoperability & Compatibility
- Accessibility solutions early in development
- Browser deployment
- Mobile content
- Semantic web





Recommended Reading





Web Standards: The What, The Why, And The How

The image shows a screenshot of the Smashing Magazine website. The header features the magazine's logo (a stylized 'S') and the text "SMASHING MAGAZINE". Below the header, there is a red banner with the word "Articles" and a small icon of a smartphone. Underneath the banner, the text "Design & development" is visible. A large, white, rounded rectangular box contains the title "Web Standards: The What, The Why, And The How".

- <https://www.smashingmagazine.com/2019/01/web-standards-guide>
Accessed September 2021





Of Patterns and Power: Web Standards Then & Now



**OF PATTERNS AND POWER: WEB
STANDARDS THEN & NOW**

- <http://www.zeldman.com/2016/01/05/13913/> Accessed August 2021





Web Standards Project (WaSP): Buzzing for Small Businesses in the Wild Web



**Web Standards Project (WaSP): Buzzing for
Small Businesses in the Wild Web**

- <https://financesonline.com/web-standards-project-wasp-buzzing-for-small-businesses-in-the-wild-web/> Accessed August 2021





Validation and SEO

- Differing opinions...

WHAT ROLE DOES W3C VALIDATION HAVE IN SEO?

<https://www.highervisibility.com/blog/what-role-does-w3c-validation-have-in-seo/> Accessed August 2021

A screenshot of a blog post from SEO Inc. The header features the SEO Inc. logo in red and white. To the right are search and menu icons. The main content area has a dark background. On the left, a vertical sidebar displays the date '24 APR 2019'. The main title 'W3C Validation Errors and How They Relate to SEO' is centered in large, bold, white letters. Below the title, the author 'GARRY GRANT' and the category 'SEARCH ENGINE OPTIMIZATION' are listed in smaller white text.

<https://www.seoinc.com/seo-blog/w3c-validation-and-seo/> Accessed August 2021





The Future Of The Web: Where Will We Be In Five Years?

- Written in 2009



The Future Of The Web: Where Will We Be In Five Years?

- <http://www.noupe.com/design/the-future-of-the-web-where-will-we-be-in-five-years.html> Accessed August 2021





Web Standards: Standards or Stasis

- Argument against web standards?
 - Note: it was last updated in 2019 but some of the discussion of web-related tech is out dated

Web Standards: Standards or Stasis



By **Rick Strahl**

Published in: **Online CODE Magazine: The Web View**

Last updated: February 22, 2019



- <https://www.codemag.com/article/060023/Web-Standards-Standards-or-Stasis> Accessed August 2021





The Trouble with Web Standards

- Another argument against web standards?
 - Note: this is an extract from an older edition of Zeldman's book



- <https://creativepro.com/the-trouble-with-web-standards/> Accessed August 2021



Process 2020

- Proposal to improve and speed up the W3C standardisation process



- <https://www.w3.org/2020/05/AC/talk/Process2020> Accessed August 2021





Additional Resources

- General websites relating to standards
 - The W3C <https://www.w3.org/>
 - WHATWG community <http://www.whatwg.org/>
 - The Web Standards Project
 - › WASP FAQ - <http://www.webstandards.org/learn/faq/>
- Mark-up validator
 - <https://validator.w3.org/>
- Device testing
 - <http://quirktools.com/screenfly>





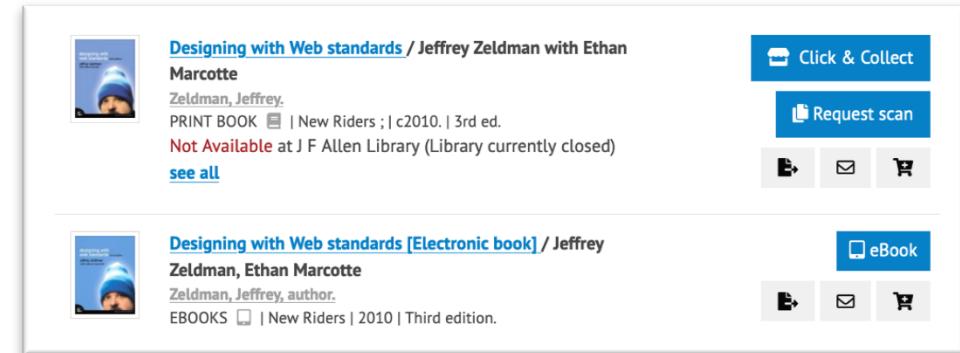
Required Reading & Consolidation





Reading & Research

- Week 1 reading
 - Designing with Web Standards by Zeldman with Marcotte (chapter 1)
 - › In the JF Allen Library (Physics)
 - › or online at https://encore.st-andrews.ac.uk/iii/encore/record/C__Rb3028858



- Week 1 research exercise
 - who uses the web, to do what, using what?
 - › <https://studres.cs.st-andrews.ac.uk/IS5103/Lectures/W01-Introduction/Exercises/W01-Exercises.pdf>





Next...

- Evolution of Mark-up





IS5103 Web Technologies: The Evolution of Mark-up

Dr Ruth Letham





Overview

- Markup Languages
- HTML Basics
- Create, Test & Publish
- XHTML Basics





Markup Languages

- Provide way to annotate documents
- Annotations are visually distinguishable from content
 - Composed of elements with attributes
- Elements mark-up content
 - Describe the purpose of the content
 - Some limited description of relationship to other elements
- Latex used for documents
- HTML & XHTML used for web pages





HTML Basics





Syntax: elements

- HTML elements are written as *tags*
 - Some come in pairs (i.e. they have open and close tags)
 - Others are standalone (i.e. they do not have a close tag)
- Open or standalone tag names are written between angled brackets < >
 - Browsers know to interpret anything between < and > as a tag name
 - Examples: <head>
- Close tag names are written between angled brackets with a slash </ >
 - Examples: </head>
- Content normally goes between open/close tags
 - Example: <p>This is a paragraph!</p>





Syntax: attributes

- Attributes are key-value pairs written within open or standalone tags
 - Provide additional information
- Some attribute can be used on any element
 - E.g. “id” (whose value is a **unique** identifier), “class” (primarily used for CSS)
- Some attributes only apply to specific elements
- Some elements must have specific attributes
- Example:
``





Document Structure

- Document type
 - HTML version
- HTML Root
 - Contains the entire page
- Head
 - Meta-data to help browser interpret and render the page
 - Title to display in the browser title bar
 - Links and relationship to other files
- Body
 - Marked-up content to display on the page

```
<!DOCTYPE html>
<html>
  <head>
    <title>My First Page</title>
  </head>
  <body>
    Some more HTML tags creating page content
  </body>
</html>
```





Head

- Head:
 - Contains tags with information about the current page
 - Not displayed on page
- Meta: data about data
 - Encoding
 - Author, description, copyright, viewport, ...
- Link:
 - Relationship to other files to be sourced *before* rendering page (e.g. CSS)
- Title:
 - Page title to display in the browser title bar





Head, Title & Meta Example

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>University of St Andrews -
      Scotland's first university, founded 1413
    </title>
    <meta name="copyright"
          content="Copyright (c) 2008 University of St Andrews" />
    <meta name="description"
          content="University of St Andrews -
            Scotland's first University" />
  </head>
  <body> ... </body>
</html>
```





Headings

- Used in the body of the page to display text
- 6 levels of heading
 - Open/close pairs with content between
 - <h1>Top level heading</h1>**
 - ...
 - <h6>sub sub sub sub sub heading!</h6>**
- Signify *purpose* of the text
 - Help give document logical structure
 - Browser determines how to display text to convey the purpose
 - › Normally in larger, bolder font

```
<h1>This is Heading 1</h1>
```

```
<h2>This is Heading 2</h2>
```

```
<h3>This is Heading 3</h3>
```

```
<h4>This is Heading 4</h4>
```

```
<h5>This is Heading 5</h5>
```

```
<h6>This is Heading 6</h6>
```





Paragraphs & emphasis

- Used in the body of the page to display text
- Open/close pairs with content between
 - <p>Some text in a paragraph</p>
 - <p>Some text worth emphasising. </p>
 - <p>Some text worth strengthening. </p>
- Again, signify *purpose* of the text
 - Browser determines how to display text to convey the purpose
 - › em: normally italicised
 - › strong: normally bold
 - Replace the old <i> and tags, which are presentational!





Special characters

- Some characters have special meaning to the browser
 - E.g. open/close angle bracket to signify tag names
- What if we want to display those characters?
 - Some have special names
 - Others, you can use ascii codes prefixed with &# suffixed with ;

Character	HTML Number	HTML Name
<	<	<
>	>	>
&	&	&
@	@	

More at: <https://ascii.cl/htmlcodes.htm>





Lists

- Used in the body of the page to display bullet points, numbered lists, definitions
- Open/close pairs to define type of list, open/close pairs to define items in list

`` Ordered List

`` Unordered List

`` List Item (used within ordered/unordered lists)

`<dl></dl>` Definition List (sometimes called *description* list)

`<dt></dt>` Definition Term (used within definition list)

`<dd></dd>` Definition Details (used immediately after definition term)





```
<h3>Ordered Lists</h3>
<ol>
  <li>First</li>
  <li>Second</li>
  <li>Third</li>
</ol>
```

Ordered Lists

1. First
2. Second
3. Third

```
<h3>Unordered Lists</h3>
```

```
<ul>
```

- First
- Second
- Third

Unordered Lists

- First
- Second
- Third

```
<h3>Definition Lists (A.K.A. <em>description</em> lists)</h3>
<dl>
```

- First Term
: First Details

Second Term
: Second Details

Third Term
: Third Details

Definition Lists (A.K.A. *description* lists)

First Term

First Details

Second Term

Second Details

Third Term

Third Details





Tables

- Used in the body of the page to display tabular data
 - Sometimes used for layout – but this is BAD PRACTICE
- Similar to lists, open/close pairs to define enclosing structure, nested open/close pairs to define contents
 - `<table>` Tabular structure
 - `<tr>` Table Row
 - `<td>` Table Data (A.K.A. a table *cell*)
 - `<th>` Instead of td – indicates the cell is a *header*
 - `<caption>` Caption to describe the contents of the table
 - `<thead> <tfoot> <tbody>` Semantic grouping of rows into header, footer and main contents





```
<table>
  <caption>Table 1: Average height and weight by dog breed.</caption>
  <thead>
    <tr> <th>Breed</th> <th>Height</th> <th>Weight</th> </tr>
  </thead>
  <tbody>
    <tr> <th>Bichon Frise</th> <td>23-30cm</td> <td>3-5kg</td> </tr>
    <tr> <th>Labrador</th> <td>54-57cm</td> <td>28-30kg</td> </tr>
  </tbody>
  <tfoot>
    <tr> <th>Overall</th> <td>38.5-43.5cm</td> <td>15.5-17.5kg</td> </tr>
  </tfoot>
</table>
```

- Notice: the browser has chosen how to display the table
 - Text styles
 - Text alignment

Table 1: Average height and weight by dog breed.

Breed	Height	Weight
Bichon Frise	23-30cm	3-5kg
Labrador	54-57cm	28-30kg
Overall	38.5-43.5cm	15.5-17.5kg





Anchors

- Used to provide *hyperlink* functionality
 - Specify a *hypertext reference* to jump to
- Open/close pairs define contents, attribute value define exact behaviour
 - The *href* attribute specifies the destination of a clickable link
 - `back to top` jumps to element with matching id in the current page
 - `page 2` loads webpage on the current site
 - `Html Checker` loads page in external website
 - `contact` opens default email client and populates 'to' field





Anchor states

- Anchors with href attribute can have states:
 - link: has not been clicked (default)
 - visited: has been clicked
 - active: mouse has been pressed (but not released)
 - hover: the mouse is over the link (or has focus)
- Browsers default display anchors differently depending on state:
 - link: blue underlined
 - visited: purple underlined
 - active: red underlined
 - hover: destination displayed in browser footer





Images

- Used in the body of the page to display images
 - Technically, image is not embedded, just linked to
 - Browser downloads image file separately and displays in image placeholder
- Standalone tag, uses *attributes* to specify image
 - Use the *src* attribute to specify the location of the image file

```

```

relative path to image on current website

```

```

absolute path to image on current website
 - Use the *alt* attribute to specify alternative text

```

```

displayed if image file cannot be found; used by user agents





Images

- Supported file formats
 - .gif, .jpeg, .png, .webp, ...
- Need to consider:
 - Screen Resolution, Size & Download
 - Compression options
- Can set a custom *favicon*
 - using a *link* in the head of the webpage

```
<link rel="icon" href="favicon.ico">
```





Finding and Using Images

- Watch out for intellectual property, copywrite and trademark restrictions
 - Only use images you have permission to use
- Creative common licence
 - https://www.google.co.uk/advanced_image_search
 - › Restrict ‘usage rights’
 - › Check licence & attribution details at source
 - <https://commons.wikimedia.org/wiki/Category:Images>
 - › Provides licence & attribution details
 - <https://www.flickr.comcreativecommons/>
 - › Browse by licence type
- Provide attribution where required





usage rights:

Creative Commons licences

all

Creative Commons licences

Commercial and other licences

flickr Sign Explore Prints Get Pro

Explore / Creative Commons

Many Flickr users have chosen to offer their work under a Creative Commons license, and you can browse or search through content under each type of license.

Use this file on the web

Page URL:
<https://commons.wikimedia.org/wiki/File:Earth.svg>

File URL:
<https://upload.wikimedia.org/wikipedia/commons/b/be/Earth.svg>

Attribution:
mehmetaergun, CC BY-SA 3.0, via Wikimedia Commons

mehmetaergun, CC BY-SA 3.0 <<http://creativecommons.org/licenses/by-sa/3.0/>>, via Wi HTML

Embed this file

HTML BBCode 512px wide

```
<a title="mehmetaergun, CC BY-SA 3.0 &lt;http://creativecommons.org/licenses/by-sa/3.0/&gt;, via Wikimedia Commons" href="https://commons.wikimedia.org/wiki/File:Earth.svg"></a>
```

IS5103 Web Tech - Markup 2022-23 © School of Computer Science



Comments

- Include text for the developer to read
 - Not displayed on the website
 - ...but visible in the source
- Not a tag

```
<!-- This is a comment for a developer to read -->
```





Create, Test & Publish





Creating a web page

- Use any plain text editor of your choice
 - Notepad++, Atom, Brackets, ...
 - Ideally pick one with colour-coded text
- Or, use an IDE
 - VS Code, Eclipse (with Web Tools Platform), ...
- Save file with .html extension





Initial testing

- Open file in a browser
 - To see rendered mark-up
- Beneath the surface
 - View Page Source from browser for static view of html
 - › Can do this for any website
 - Use browser web tools to ‘inspect’ html for a more interactive view of html
 - › Identify mark-up in use
 - › Can help identify errors in tags: wrong nesting, missing/extraneous open/close tags, etc.





Publishing

- Recall:
 - Web pages are hosted on a server
 - Web server software looks in specified folders based on HTTP request
- On the School system:
 - Every cs user has their own domain e.g.
 - › <https://mucs.host.cs.st-andrews.ac.uk>
 - Each user's domain links to the `nginx_default` folder in their home directory
 - Any files in your `nginx_default` folder can be accessed via the web e.g.
 - https://mucs.host.cs.st-andrews.ac.uk/my_first_page.html
 - › Will serve the `my_first_page.html` file in `/cs/home/mucs/nginx_default` folder





Publishing: Documentation

- For general information on remote working
 - https://systems.wiki.cs.st-andrews.ac.uk/index.php/Working_remotely
 - I recommend the video tutorials linked from the remote working page
- For specific information on how to access your CS home
 - https://systems.wiki.cs.st-andrews.ac.uk/index.php/How_to_use_the_Home_service
 - Choose the link for your own operating system
- For information on how to set up your nginx_default folder
 - https://systems.wiki.cs.st-andrews.ac.uk/index.php/New_Linux_Web_Service#Serving_static_HTML_files





Further Testing

- Consider appearance, usability and functionality
- View the page in different browsers
- Check links and images
 - e.g. <http://validator.w3.org/checklink>
- Simulate different devices
 - e.g. <https://bluetree.ai/screenfly/>
- Use a mark-up validator
 - e.g. <https://validator.w3.org/>
- Use a checklist
 - E.g. <https://www.softwaretestinghelp.com/web-application-testing/>





XHTML

January 2000





What is XHTML?

- eXtensible Hyper Text Markup Language
 - Recommendation developed & controlled by the W3C
- Reformulation and combination of XML and HTML 4
 - Subset *and* extension of HTML 4
- Developed to address issues with
 - Increasing use of **proprietary** HTML
 - New and evolving browsing technologies
 - Interoperability with alternative user agents
- Combined strengths of HTML and XML to
 - Encourage *structural* and *meaningful* mark-up
 - Discourage presentational hacks





HTML vs. XHTML

XHTML Rules

- Has mandatory elements
- Tags must be properly nested
- Tag & names must be lower case
- Paired tags must be explicitly closed
- Standalone tags must ‘self close’
- Attributes must be in quotes
- Attributes minimization is forbidden
- Name attribute is deprecated

Example

html, head, title and body

`<div><p>text</div></p>` is forbidden

`` is forbidden

`<p>text 1<p>text 2` is forbidden

`` is forbidden

`` is forbidden

`` is forbidden

`` is forbidden





DOCTYPE

- First line in XHTML document
- Specifies a Document Type Definition (DTD)
 - Specifies permitted syntax and grammar in Standard Generalised Markup Language (SGML)
- Required have for validation purposes!





DTD

- Defines the legal structure, elements and attributes that are available for use
- XHMTL has a STRICT version
 - <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
 - Does not allow deprecated elements or attributes
 - Does not allow presentational elements
- XHMTL has a TRANSITIONAL version
 - <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
 - Allows some presentational elements
 - Allows some deprecated elements and attributes
 - Used to support early browser versions





Validation

- We have already seen the W3C validator
<https://validator.w3.org/>
- Documents are verified against the declared DTD
 - DTD must be added as the first line of the file
- Check structure and use of elements and their attributes are consistent with the definitions in the DTD
- Check conformance to W3C recommendations and other standards
- Should make your web site faster loading, more search engine friendly, accessible to a greater number of devices
 - Think of it as proof-reading your code for mistakes





XHTML Validation – Experience

- Go to <https://validator.w3.org/>
- Select ‘Validate by Direct Input’ tab
- Expand ‘More Options’
- Make sure ‘Only if Doctype is missing’ is checked
- Paste your XHTML into the text area
- Click ‘Check’ button

The screenshot shows the W3C Markup Validation Service interface. The browser title bar reads "W3 The W3C Markup Validation Service". The address bar shows the URL "https://validator.w3.org/#validate_by_input+with_options". The tabs at the top are "Validate by URI", "Validate by File Upload", and "Validate by Direct Input", with "Validate by Direct Input" being the active tab and circled in red. Below the tabs is a section titled "Validate by direct input" containing an example of XHTML code. Underneath is a "More Options" section with a blue arrow pointing down, also circled in red. This section contains two radio buttons: "Validate Full Document" (selected) and "Validate HTML fragment". Under "Validate Full Document", there is a dropdown menu for "Use Doctype" set to "(detect automatically)" and a checked checkbox "Only if Doctype is missing". Other options in the "More Options" section include "Validate HTML fragment", "List Messages Sequentially" (selected), "Show Source", "Show Outline", "Clean up Markup with HTML-Tidy", "Validate error pages", and "Verbose Output". At the bottom right is a "Check" button.



XHTML Validation – Experience

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<head>
<title> The Use of Acronyms</title>
<body>
<h1> HTML & XHTML Mark-up</h1>
<h2> WCAG2.0</h2>
<h3>CSS Cascading Style Sheets</h3>
<HR>
There are many <strong> <em> acronyms </strong>
> </em> associated with the WWW.
<a href="http://www.w3schools.com">visit W3Schools.com!>

</body>
</html>
```

- You can edit and ‘Revalidate’

The screenshot shows a browser window displaying the W3C Markup Validation Service. The URL is https://validator.w3.org/check. The page title is "Markup Validation Service". A red banner at the top states "Errors found while checking this document as XHTML 1.0 Strict!". Below it, the "Result" section shows "17 Errors, 3 warning(s)". The "Source" section contains the original XHTML code. The "Validation Output" section lists 17 errors, starting with:

- Line 2, Column 6: document type does not allow element "head" here; assuming missing "html" start-tag
<head>
- Line 2, Column 1: Missing xmlns attribute for element html. The value should be: http://www.w3.org/1999/xhtml
<head>

At the bottom, a note explains that many XML-based documents need a mandatory `xmlns` attribute on the root element, with an example for XHTML.



Round up

- XHTML is a subset and extension of HTML 4.0
- XHTML has more restrictive rules
 - Including more separation of content and meaning from presentation
 - And not allowing content managers to add their own elements to the language
- DTD provides mechanism to allow the creation of language subsets and the addition of extensions
 - XHTML Basic, MathML,





XHTML 2.0

- Proposed successor to XHTML 1 (2009-2010)
- What happened?
 - Corrected semantics but without offering new features
 - Backwards incompatible with previous mark-up languages
- Migration to HTML5 deemed easier by many





HTML Version Statistics: then

2008

- XHTML 1.0 ~60%
- HTML 4.01 ~20%
- HTML 5 ~0%
 - First release January 2008
 - W3C Recommendation October 2014

September 2012

- XHTML 1.0 > 50%
- HTML 4.01 ~10%
- HTML 5 > 15%



<http://try.powermapper.com/demo/statsversions.aspx>
accessed September 2012

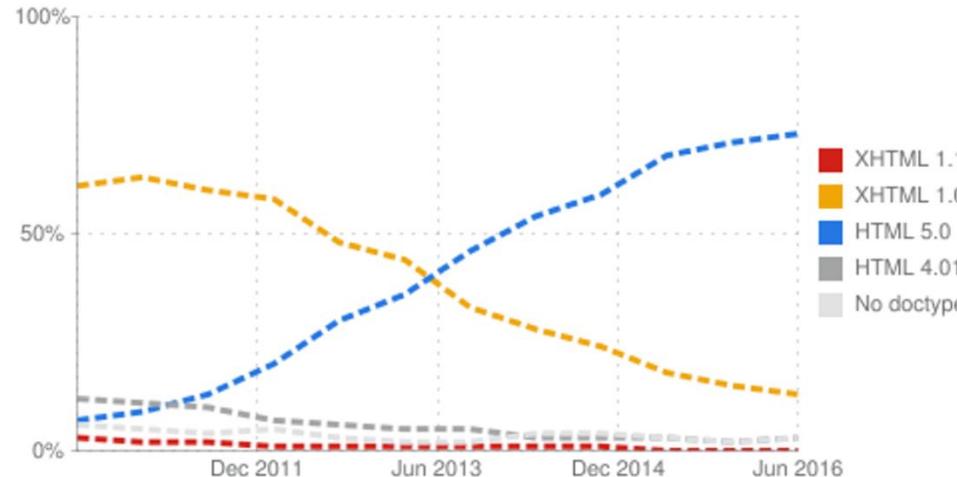




HTML Version Statistics: now

September 2016

- XHTML 1.0 < 15% 
- HTML 4.01 < 3% 
- HTML 5 > 70% 



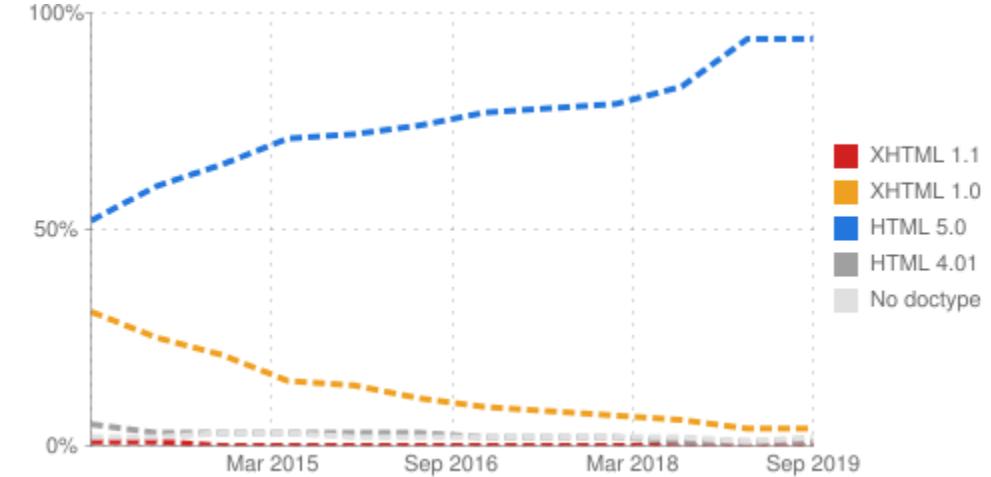
<http://try.powermapper.com/Stats/HtmlVersions>

accessed September 2016

IS5103 Web Tech - Markup

August 2022

- XHTML 1.0 < 5% 
- HTML 4.01 < 1% 
- HTML 5 > 90% 



<http://try.powermapper.com/Stats/HtmlVersions>

accessed August 2022

2022-23

© School of Computer Science





Recommended Reading





Why tables for layout is stupid

Why tables for layout is stupid:

problems defined, solutions offered

Tables existed in HTML for one reason: To display tabular data. But then border="0" made it possible for designers to have a grid upon which to lay out images and text. Still the most dominant means of designing visually rich Web sites, the use of tables is now actually interfering with building a better, more accessible, flexible, and functional Web. Find out where the problems stem from, and learn solutions to create transitional or completely table-less layout.

- <https://www.hotdesign.com/seybold/> Accessed August 2021





Validation and SEO

- Re-read article from last week

WHAT ROLE DOES W3C VALIDATION HAVE IN SEO?

<https://www.highervisibility.com/blog/what-role-does-w3c-validation-have-in-seo/> Accessed August 2021

- Look at the date it was published and think about the article in the context of XHTML





Required Reading & Consolidation





Reading & Consolidation

- Week 2 reading
 - Designing with Web Standards by Zeldman with Marcotte (chapter 5)
 - › In the JF Allen Library (Physics)
 - › or online at https://encore.st-andrews.ac.uk/iii/encore/record/C__Rb3028858

The screenshot shows a library catalogue interface with two search results for 'Designing with Web standards'.
1. **Physical Book Result:**
 - Title: [Designing with Web standards](#) / Jeffrey Zeldman with Ethan Marcotte
 - Author: [Zeldman, Jeffrey](#)
 - Type: PRINT BOOK
 - Availability: Not Available at J F Allen Library (Library currently closed)
 - Buttons: Click & Collect, Request scan, Print, Email, Cart.
2. **Electronic Book Result:**
 - Title: [Designing with Web standards \[Electronic book\]](#) / Jeffrey Zeldman, Ethan Marcotte
 - Author: [Zeldman, Jeffrey, author.](#)
 - Type: EBOOKS
 - Availability: Available at J F Allen Library (Library currently closed)
 - Buttons: Click & Collect, Request scan, Print, Email, Cart.

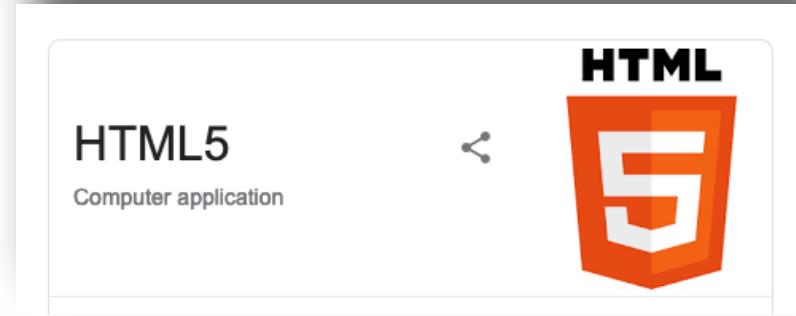
- Week 2 consolidation exercise
 - Writing HTML
 - › <https://studres.cs.st-andrews.ac.uk/IS5103/Lectures/W02-Markup/Exercises/W02-Exercises.pdf>





Next...

- Migration to HTML5
- Introduction to CSS





IS5103 Web Technologies: HTML5 & Intro to CSS

Dr Ruth Letham





HTML5

Document structure with semantic tags





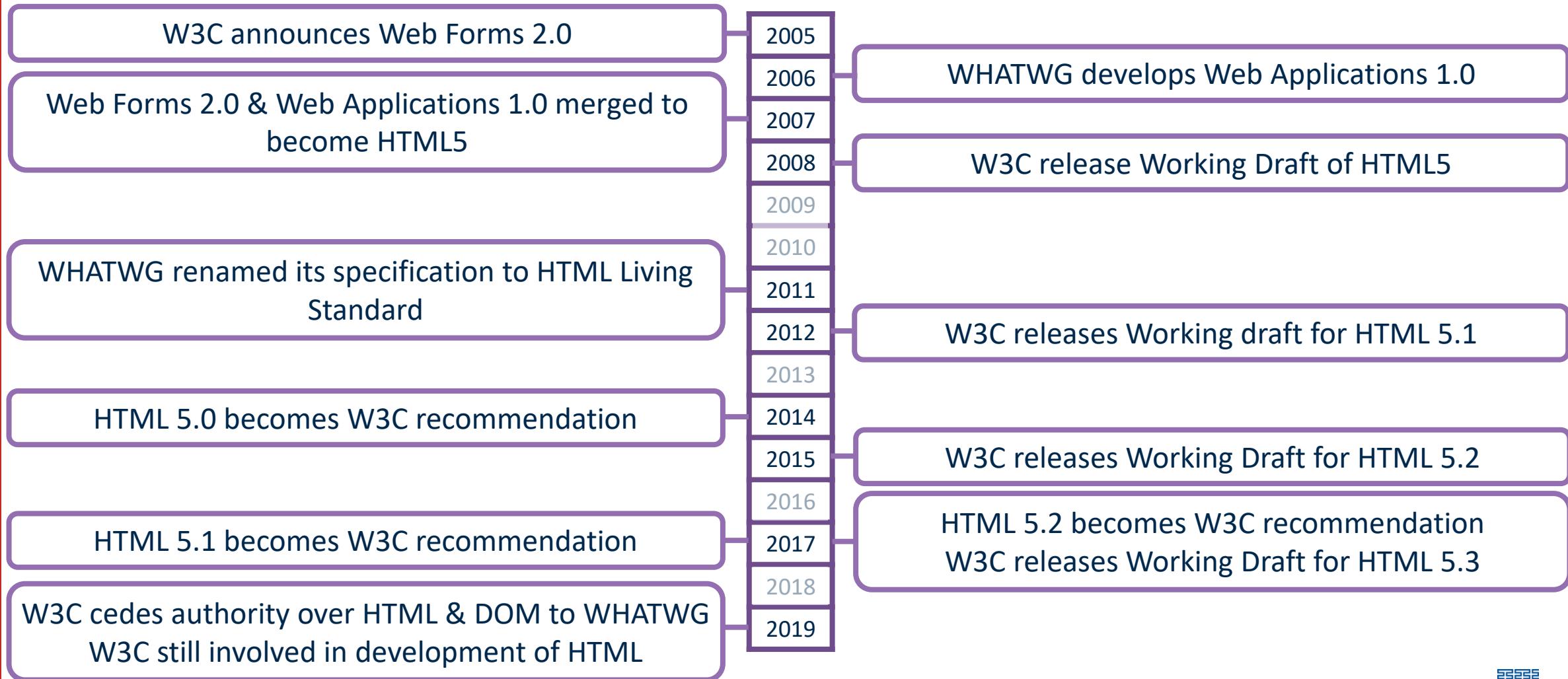
Overview

- HTML5
 - Timeline
 - Syntax
 - Document structure
 - Page & text structure
 - Multimedia
 - Forms
- Markup Examples
- Browser Compatibility





HTML5 Timeline





HTML5

- HTML Living Standard is now the authoritative HTML5 standard
 - <https://html.spec.whatwg.org/>
- Collection of features not a new mark-up language
- Builds upon HTML 4.01 and XHTML 1.0
- New, redefined & deprecated elements
 - Parsed over a billion pages to look at id and classes in use
- Improved support for forms & media
- Shift from information sharing toward interactive systems

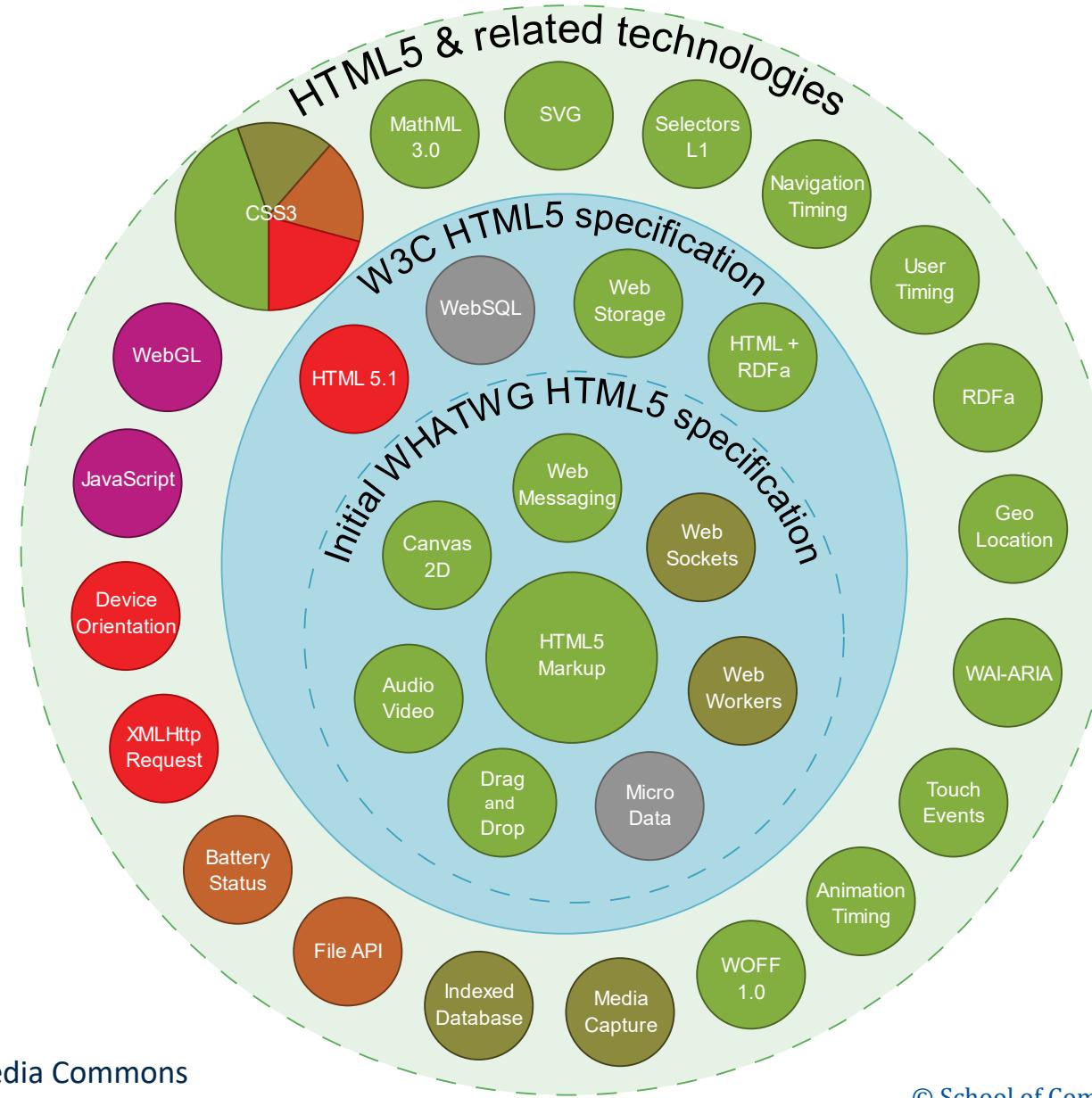




HTML5

Taxonomy & Status (October 2014)

- Recommendation/Proposed
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Deprecated or inactive



[Mercury999](#), CC BY-SA 4.0, via Wikimedia Commons

© School of Computer Science





HTML5 Syntax & Semantics

- HTML5 has a language specification to define permitted elements, attributes, values, etc.
- HTML5 elements, attributes, and values have specific *meanings*
 - Unlike XHTML, which had meaning only for outline structural elements (head, title, body)
 - Allows documents & apps to be used in a wide variety of contexts & user agents
- HTML5 does not prescribe a writing style
 - Unlike XHTML, anything goes: uppercase, lowercase, self-closing or not...
 - Enforce your own writing style
 - Apply consistency for maintainability
- <https://html.spec.whatwg.org/multipage/#toc-semantics>





HTML5 Outline Structure

- Doctype
 - No version number
 - No DTD
 - Simplified
- HTML Root
 - Language
- Head
 - Title, metadata, relationships
- Body
 - Content

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8"/>
    <title>My Page</title>
    <link rel="stylesheet" href="style.css"/>
  </head>
  <body></body>
</html>
```





HTML5 Sections

- <article>
 - Content makes sense as a standalone piece
- <section>
 - Core content that needs to be in context
 - Makes sense to have a heading
- <nav>
 - Navigational component
- <aside>
 - Non-core content
- <address>
 - Contact information
- <header>
 - Introductory and/or navigational aids
- <footer>
 - Information about the page/section it is in, e.g. author, copyright, appendices, etc.
- <hgroup>
 - Grouping of headings
 - Used by agents when building a document outline

Full listing in 4.3 Sections at <https://html.spec.whatwg.org/multipage/#toc-semantics>

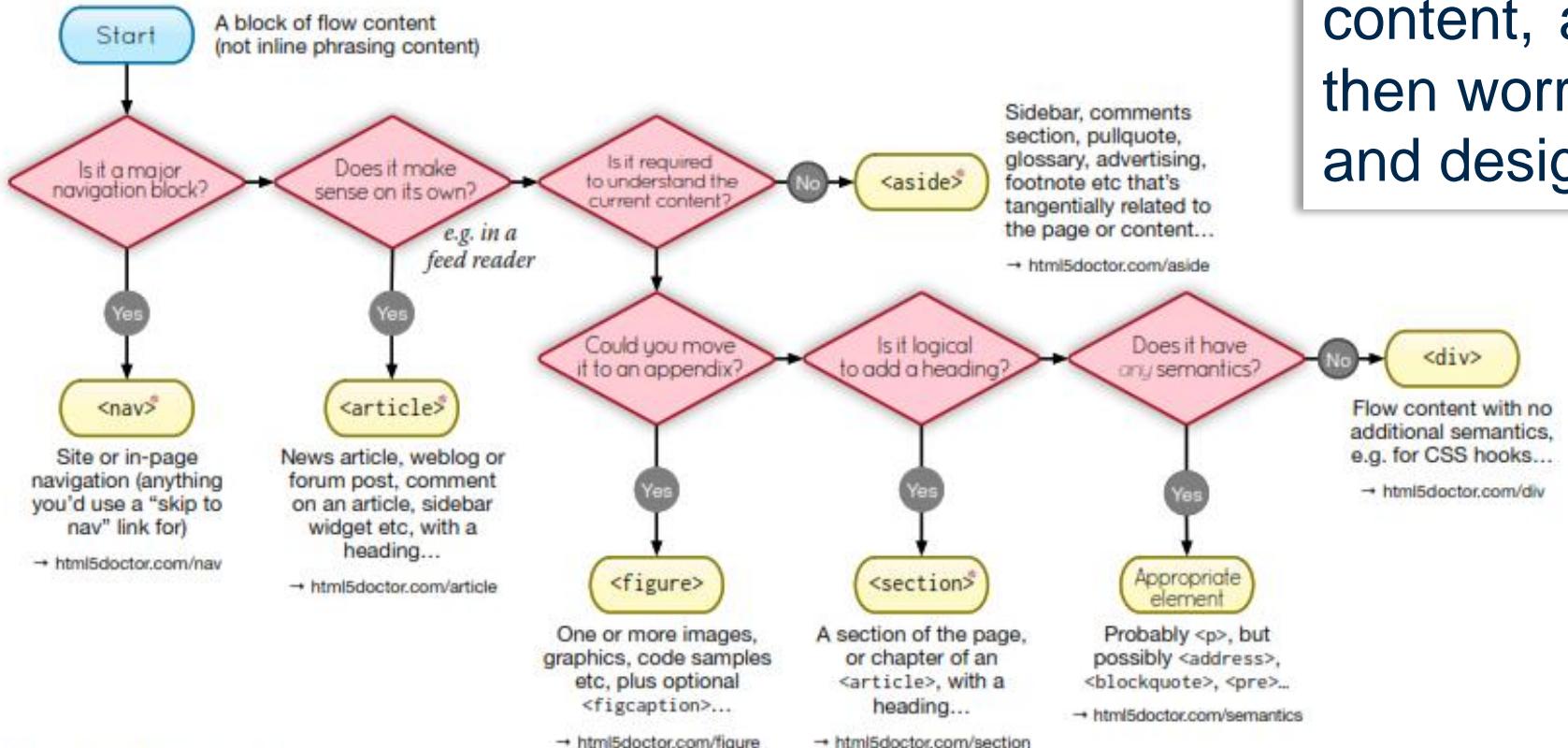




HTML5 Element Flowchart

Sectioning content elements and friends

By @riddle & @boblet
www.html5doctor.com



Choose elements that best describe the ***meaning*** of your content, and how they relate, then worry about presentation and design.





Grouping

- <menu>
 - Semantic alternative to unordered list
 - Not widely supported (yet)
- <figure>
 - Content that is self-contained and is referenced as a single unit
 - Often used for illustrations, diagrams, photos, code listings, etc.
 - Often with a caption
- <figcaption>
 - Caption or legend for a figure
 - First or last element within a figure element

Full listing in 4.4 Grouping content at <https://html.spec.whatwg.org/multipage/#toc-semantics>





Text-level Semantics

- <cite>
 - Content is the title of cited work
- <abbr>
 - Content is an abbreviation
 - Title attribute is the expansion (optional)
- <time>
 - Content is a date/time string in a recognized format
- <mark>
 - Content is marked or highlighted

Full listing in 4.5 Text-level semantics at <https://html.spec.whatwg.org/multipage/#toc-semantics>





Deprecated/obsolete Elements

- Mostly presentational tags...
- <center>
-
- <tt>
- <acronym>
- <strike>
- <frame>
- <frameset>





Multimedia

4.8 Embedded content

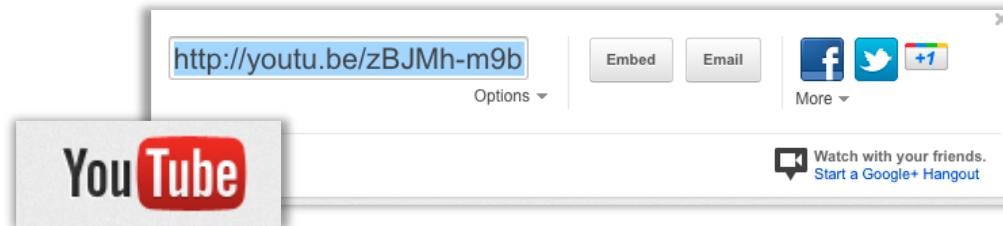
<https://html.spec.whatwg.org/multipage/#toc-semantics>





Then...

- No standards-based way to embed media
- Browser Plug-ins
- Video or audio player
- Flash Player, Silverlight, Quicktime & Realplayer
- 3rd Party Service





Now...

- Video for everyone!
 - Open standards
 - Can use script to alleviate browser problems
 - › But make sure you don't fall back onto defunct tech: e.g.
http://camendesign.com/code/video_for_everybody
- Containers & Codecs for video and audio





Now...

- <video>
 - Video, movie, or audio files with captions
- <audio>
 - Sound or audio stream
- <track>
 - External timed text tracks for media elements (e.g. subtitles, captions, etc.)
 - Does not represent anything on its own





Audio & Video

- Attributes used for content and configuration
- **src** – URI of the resource
 - Alternatively, you could use the `<source>` element
- **crossorigin** – how to handle requests for different origins
- **preload** – how much buffering the resource will likely need
- **autoplay** – whether it can be started automatically on page load
- **loop** – whether to loop playback
- **muted** – whether to mute the media resource by default
- **controls** – whether to show user agent controls





Video

- Video has some extra attributes that do not apply to audio
- **poster** — frame to show prior to video playback
- **playsinline** — display video content within the element's playback area
- **width** — horizontal dimension
- **height** — vertical dimension
- Note: user agents may ignore/overrule some of the audio/video settings





Track

- Child element of an audio/video element
- Attributes used for content and configuration
 - **kind** — The type of text track
 - **src** — Address of the resource
 - **srlang** — Language of the text track
 - **label** — User-visible label
 - **default** — Enable the track if no other text track is more suitable





Forms

4.10 Forms

<https://html.spec.whatwg.org/multipage/#toc-semantics>





Forms

- Allow data to be sent to a server for processing
 - We will cover the in detail later in the course!
- Basic structure:
 - A form element with attributes describing where and how to post the data
 - Multiple `input`, `select`, `textarea`, `button` elements to gather user input
- Then:
 - Validation & restrictions on types of data entered done in JavaScript
- Now:
 - Automatic input validation & more input types in HTML





Form Controls & Validation

- Can specify restrictions to the *type* of data permitted in an input element
 - tel, url, email, date, month, week, time, datetime-local, number, range, color, image, search, ...
- Some essentially just add validation
 - E.g. email, url, ...
- Others may change the display
 - E.g. date pickers, number spin boxes/sliders, colour pickers, ...
- Attributes provide more control
 - e.g. pattern, required, autocomplete, list, placeholder





Markup Examples

The Good & the Bad





Rotten Mark-up

```
<table border="0" cellspacing="0" cellpadding="0" align="center">
  <tr valign="bottom">
    <td align="center" height="33"><a href="/102-8084626-4815355"></a>&ampnbsp</td>
    <td>
      <table border="0" cellspacing="0" cellpadding="0" align="center" class="tabs">
        <tr id="twotabtop">
          <td class="lefton" height="33"> <div align="center">
            <a href="/102-8084626-4815355"></a>
          </div></td>
          <td class="middleoffonleft" height="33"> <div align="center">
            <a href="/102-8084626-4815355"><span style="text-transform: none;">Your <br/>Amazon.com</span></a>
          </div></td>
          <td class="middleoff" height="33"> <div align="center">
            <a href="/102-8084626-4815355" name="two-tabs|he|all-categories">See all 35<br/>Product&ampnbspCategories</a>
          </div></td>
          <td class="rightoff" width="15" height="33">
            
          </td>
        <tr class="bottom">
          <td class="tools" nowrap="nowrap" height="33" align="right">&ampnbsp&ampnbsp<a href="/102-8084626-4815355" >Your Account</a>
            <span class="navspacer"><span class="light">&#124;</span></span>
            <a href="/102-8084626-4815355"> Cart</a> ...
          </td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```





Meaningful Mark-up

```
<section class="intro" id="zen-intro">
  <header role="banner">
    <h1>CSS Zen Garden</h1>
    <h2>The Beauty of <abbr title="Cascading Style Sheets">CSS</abbr> Design</h2>
  </header>

  <div class="summary" id="zen-summary" role="article">
    <p>A demonstration of what can be accomplished through <abbr title="Cascading Style Sheets">CSS</abbr> ...</p>
    <p>Download the example <a href="/examples/index" title="This page's source HTML code, not to be modifie ...</p>
  </div>

  <div class="preamble" id="zen-preamble" role="article">
    <h3>The Road to Enlightenment</h3>
    <p>Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible <abbr title="...</p>
    <p>We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of ...</p>
    <p>The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to se ...</p>
  </div>
</section>

<div class="main supporting" id="zen-supporting" role="main"> ...
```

<http://www.csszengarden.com/> Accessed August 2021





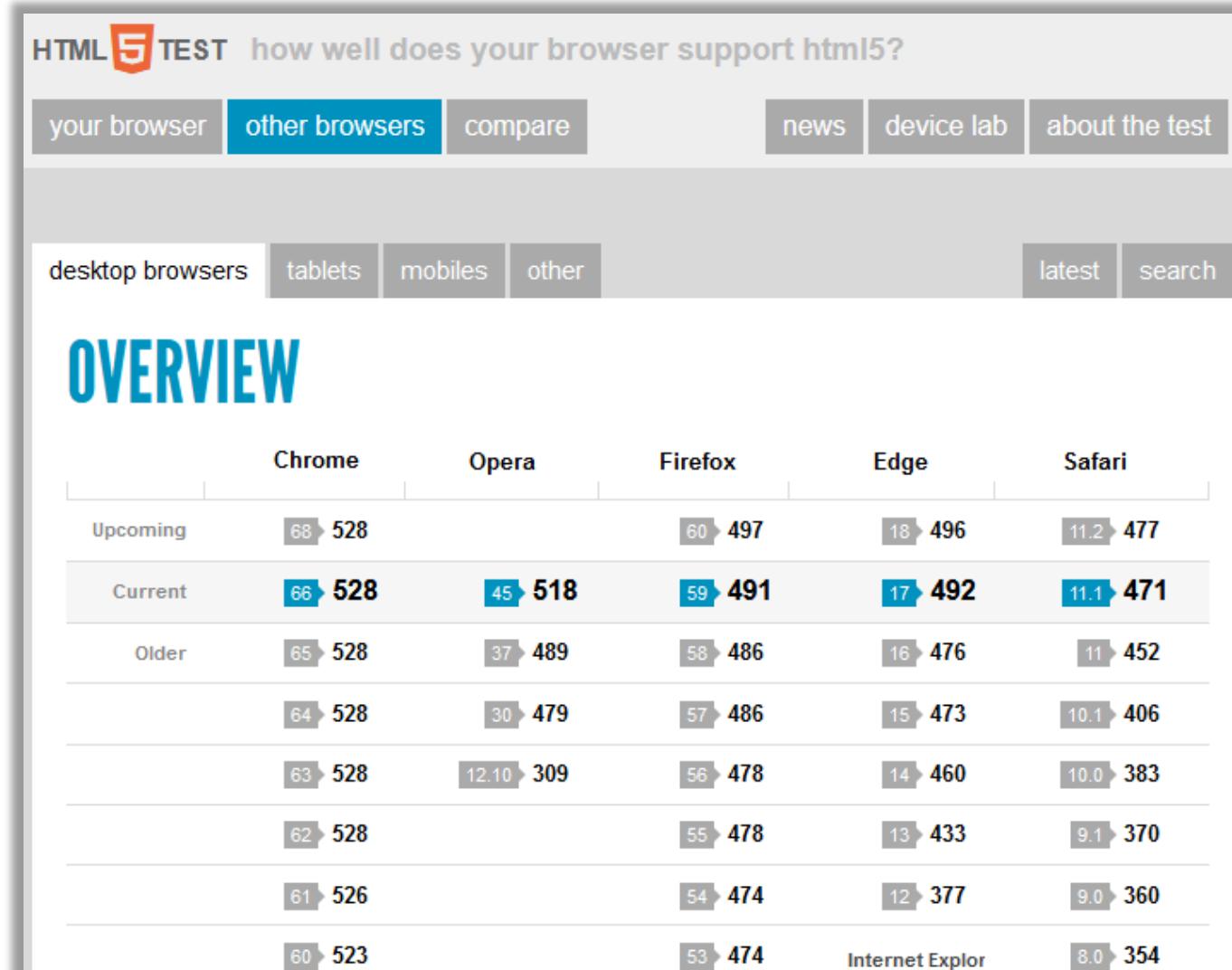
Browser Compatibility





HTML5 Compatibility

- Score out of 555 points
 - <http://html5test.com/>
- Chrome comes out on top
 - But only just...
 - Others are catching up





Quote

“Implementations and specifications have to do a delicate dance together.

You don’t want implementations to happen before the specification is finished, because people start depending on the details of implementations and that constrains the specification.

However, you also don’t want the specification to be finished before there are implementations and author experience with those implementations, because you need the feedback.

There is unavoidable tension here, but we just have to muddle on through.”

Mozilla developer 2010

<http://lists.w3.org/Archives/Public/public-html/2010Jan/0107.html> Accessed August 2021



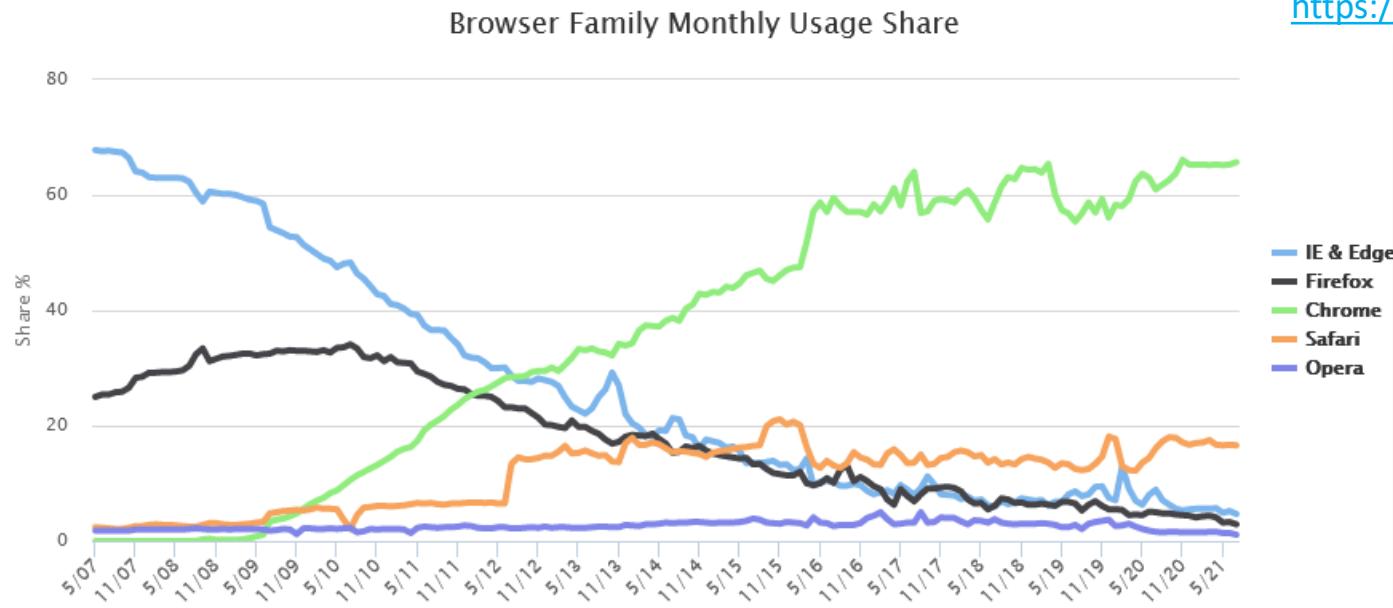


Browser Market Share

- Exact figures vary but Chrome comes out on top across the board



<https://gs.statcounter.com/> Accessed August 2021



<https://www.w3counter.com/trends> Accessed August 2021

<https://www.w3schools.com/browsers/default.asp>
Accessed August 2021

2021	Chrome	Edge	Firefox
July	81.6 %	6.0 %	5.6 %
June	81.7 %	5.9 %	5.6 %
May	81.2 %	5.8 %	5.8 %
April	80.7 %	5.6 %	6.1 %
March	80.8 %	5.5 %	6.3 %
February	80.6 %	5.4 %	6.6 %
January	80.3 %	5.3 %	6.7 %



The Browser Monopoly

The Browser Monopoly

Blair · August 20, 2019

<http://blairreeves.me/2019/08/20/the-browser-monopoly/> Accessed August 2021





CSS

Document presentation





Overview

- What is CSS?
 - ...and why do we need it?
- CSS syntax
 - Rules
 - Properties
 - Selectors
- Handling conflicts
 - Choosing from multiple applicable rules





What is CSS?

- “Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents.”
 - <https://www.w3.org/Style/CSS/Overview.en.html>
- Allow web authors control over all aspects of presentation
 - Colours, typography, size, placement, spacing, ...
 - Now also includes sophisticated animations
- Intended to allow separation of content and presentation
- Evolving to addresses weakness in presentation controls
 - CSS to replace HTML table-based layouts, frames and other presentational hacks





Remember: Separation of concerns

The screenshot illustrates the CSS Zen Garden website, demonstrating the separation of concerns between presentation and content.

Left Window (Home Page):

- Header:** CSS ZEN GARDEN, *The Beauty of CSS Design*
- Text:** A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.
- Links:** Download the example [HTML FILE](#) and [CSS FILE](#)
- Section:** THE ROAD TO ENLIGHTENMENT
- Text:** Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.
- Text:** We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WASP, and the major browser creators.
- Footer:** CSS Zen Garden

Right Window (Mid-Century Modern Design):

- Header:** CSS ZEN GARDEN
- Title:** The Beauty of CSS Design
- Text:** Select a Design:
- Options:**
 - Mid Century Modern by Andrew Lohman
 - Garments by Dan Mall
 - Steel by Steffen Knoeller
 - Apothecary by Trent Walton
 - Screen Filler by Elliot Jay Stocks
 - Fountain Kiss by Jeremy Carlson
 - A Robot Named Jimmy by meltmedia
 - Verde Moderna by Dave Shea

Code Editor (Bottom):

```
51 <body id="css-zen-garden">
52 <div class="page-wrapper">
53
54   <section class="intro" id="zen-intro">
55     <header role="banner">
56       <h1>CSS Zen Garden</h1>
57       <h2>The Beauty of <abbr title="Cascading Style Sheets">CSS</abbr> Design</h2>
58     </header>
59
60     <div class="summary" id="zen-summary" role="article">
61       <p>A demonstration of what can be accomplished through <abbr title="Cascading Style
62       <p>Download the example <a href="/examples/index" title="This page's source HTML cod
63     </div>
64
```

CSS Zen Garden
<http://www.csszengarden.com>
Accessed August 2021

© School of Computer Science



Evolution of Markup & CSS

- Original purpose of HTML was to semantically describe content
 - Simple page structure (head, body, h1 .. h6, p, div,...)
- Browsers used element properties & defaults in display
 - Block elements (appear on their own line): h1 .. h6, p, div ...
 - Inline elements: a, span ...
 - Semantic display: headings have larger font, anchors underlined and blue, ...
- Addition of presentational elements & attributes mixed style with structure
 - font, i, b, strike, bgcolor, align, etc.
 - and later proprietary elements added to this e.g. blink, marquee





Evolution of Markup & CSS

- HTML 3 draft included style tags and class attribute
 - To allow the way elements look to be redefined
 - Encourages separation of style from structure
 - Draft never made it to recommendation status but was partially implemented by most browsers
- HTML 4 included support for external style sheets
 - Encourage further separation of style from structure
 - Allow reuse of styles on multiple pages





Linking CSS to Markup

- You can use the *style* attribute to link style rules to an element
 - Known as *inline* styles:
`<p style=""></p>`
- You can use the *style* element to specify style rules for target elements
 - Known as *internal* or *embedded* styles:
`<style> </style>`
- You can *link* to separate files containing style rules for target elements
 - Known as external stylesheet:
`<link rel="stylesheet" href="example.css"/>`
- You can *import* files containing style rules for target elements
 - Used within css files or style elements:
`@import url(example.css)`
- To get the full benefit of CSS you should *link* or *import* css files





Benefits of CSS

- Global changes take minutes
 - One style sheet thousands of pages
 - Easier site maintenance
- Dynamic prototyping
 - More agile development
 - Reduce design & development time
- Bandwidth
 - Smaller documents
 - Faster page load
- Interoperability by adhering to standards
 - Increases accessibility
 - Removes presentational elements from mark-up





CSS Specifications

- CSS level 1 (1996) Core Capabilities
 - Fonts, margins, colours etc.
- CSS level 2.1 (2006)
 - Positioning, media specs etc.
- CSS level 3 (on going)
 - Split across multiple *modules*
 - Borders, vertical text, user interaction, multiple columns, ...
- Specifications and drafts:
 - <http://www.w3.org/Style/CSS/current-work>





CSS Syntax





CSS basic syntax

- A series of rules comprising
 - What element(s) the rule applies to
 - What visual display properties the rule is controlling and how
- More precisely, a rule is a selector followed by a semi-colon separated list of key-value pairs enclosed in curly braces:

```
selector { property1: value1; property2: value2; }
```

- Example:

```
p { color: #C210A0; } ■
```





CSS properties

- Cover nearly all aspects of display
- Layout & positioning
 - Type: block, inline, and more...
 - Size: width, height, font-size
 - Position: relationship to other elements, x/y, offsets
 - Spacing: additional space inside or outside the bounds of the display
- Look-and-feel
 - Font: style, family, weight, colour, shadow
 - Colour: text, background, border, opacity
 - Border: colour, width, style, rounded edges





Examples

```
position: absolute;  
bottom: 0;  
right: 0;  
font-weight: bold;  
color: cornflowerblue;  
background-color: cornsilk;  
border: thick dashed blue;  
border-radius: 10%;  
margin: 10pt;  
padding: 10pt;
```

```
border: medium solid #000;  
position: relative;  
width: 20%;  
min-height: 100pt;  
padding: 10px 40px;
```

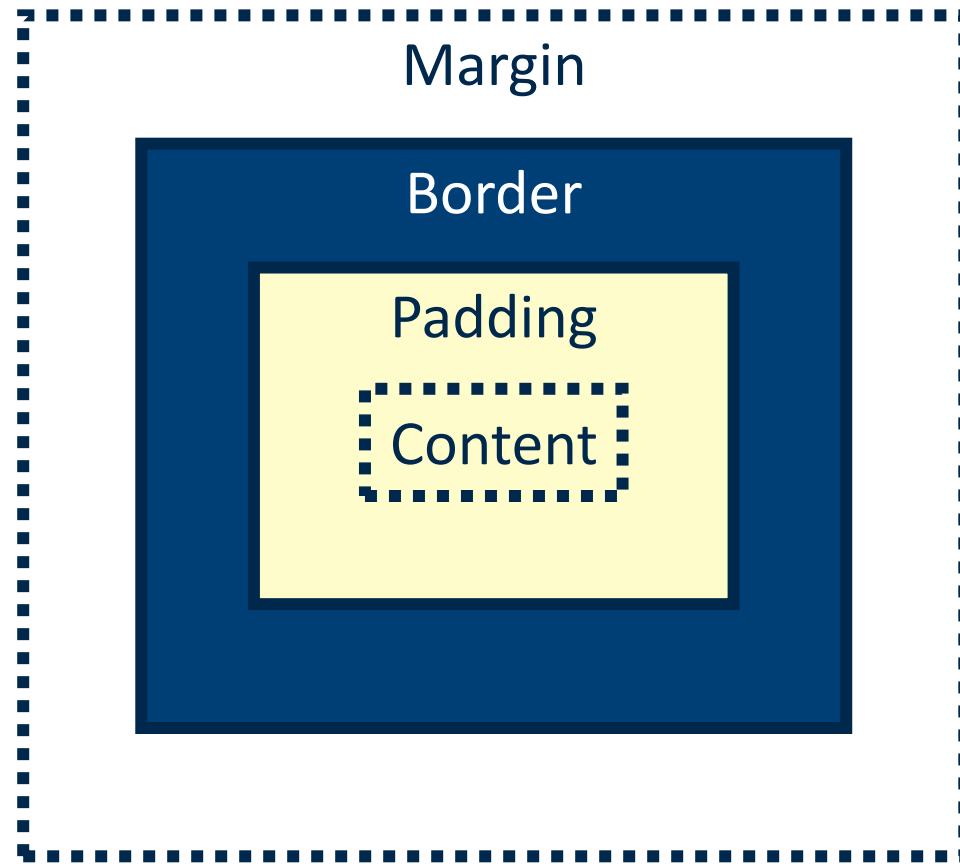
Far too many to list all!





CSS Box Model

- Each block element has an associated box
- Each box has:
 - Content: to be displayed in the element
 - Padding: edge that surrounds the content; included in the background
 - Border: stylable, visible element surrounding the padding
 - Margin: transparent edge that surrounds the border
- Can be manipulated using CSS style rules





CSS basic selectors

- An **tag** name to change the look of all instances of that element
 - e.g. `body`, `p`, `header`
- A **class** name (preceded by a dot) to change the look of all elements with that value for their class attribute
 - e.g. `.warning`, `.author`
 - Should be meaningful name that describe *purpose*
- An **id** (preceded by a hash) to change the look of a single element with that id
 - e.g. `#mission_statement`, `#top_bar`





Examples

```
aside { color: red; }
```

```
#info { color: lightblue; }
```

```
.highlight { background-color: yellow; }
```





Grouping

- What if multiple elements should have the same look-and-feel?
 - Don't want to define the same rules in multiple places ☹
- You can specify a style applies to multiple different elements by separating them with commas ☺
- Example
 - `div, .highlight`
 - Will apply the style to all div elements and all elements with the class *highlight*





Combinators

- What if you want to style elements based on where they are in the DOM?
- You could give elements a class name
 - ... but that becomes onerous and difficult to maintain ☹
 - What if you forget? Or if you need to move some elements?
- You can use combinators ☺
 - .. to style descendants (direct or indirect), siblings (adjacent or not)





Combinators: child

- Match targets only if it is a direct descendant of the first selector
- Use a greater than sign to separate parent and target
 - parent > target
- Example
 - `.info > p { color: greenyellow; }`
 - Will change the colour of any paragraph that is a direct child of an element with the class *info*





Combinators: descendant

- Match targets only if it has an ancestor matching the first selector
 - Ancestor does not need to be the immediate parent
 - Could be parent, parent's parent, parent's parent's parent, ...
- Use a space to separate ancestor and target
 - ancestor target
- Example
 - `.info p { color: greenyellow; }`
 - Will change the colour of any paragraph that is a descendant of an element with the class *info*





Combinators: descendant [2]

- **Important:** if you omit the space you change the meaning of the selector
- **Without** a space you are giving more precise target information *not* specifying a descendant
- Example
 - `div .highlight`
 - (with a space) means an element with class *highlight* that is a descendant of a div
 - `div.highlight`
 - (without a space) means a div with class *highlight*





Combinators: siblings

- Match target that has the same parent as another
- Use a ~ to separate sibling from target
sibling ~ target
- Use a + if the target has to be next to it's sibling
sibling + target
- Example
 - .info + p { color: greenyellow; }
 - Will change the colour of any paragraph that has the same parent as an element with the class *info* and is immediately next to it in the DOM





CSS attribute selectors

- What if you want to style elements based on an attribute other than id or class?
- Select target based on the existence of an attribute
 - Attribute name enclosed in square brackets
 - e.g. [**data-value**]
- Select target based on the value of an attribute
 - Attribute name and value enclosed in square brackets
 - e.g. [**alt="logo"**]





CSS pseudo classes

- What if you want more control over which elements to select?
- Select a target based on ‘extra’ information
 - Pseudo-classes
- Selector is followed immediately by a colon and the name of the pseudo class
- Can be dynamic, e.g. how a user has interacted with it
 - E.g. `:visited`, `:hover`, `:focus`
- Can be structural, e.g. where it is in relation to other elements
 - E.g. `:first-child()`, `:nth-child()`, `:nth-of-type()`
 - Some can take arguments to select a particular pattern e.g. `:nth-child(odd)`





Examples

```
tr:nth-child(odd) { background-color: aliceblue; }
```

- Will apply the style to every row that is an odd numbered child of its parent

```
td:nth-of-type(3n) { color: orangered; }
```

- Will apply the style to every third table cell in a table row

×	1	2	3	4	5
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20
5	5	10	15	20	25





CSS pseudo elements

- What if wanted *even more* control over which elements to select?
- Select *part* of a target that would not normally be accessible
 - Pseudo-elements
- Selector is followed immediately by 2 colons and the name of the pseudo element
- Access structural parts
 - E.g. `::first-line`, `::first-letter`, `::before`





Examples

```
.longtext::first-line { text-transform: uppercase; }
```

- Will apply the style to the first line (as displayed by the browser) of any element with the class *longtext*

```
a::before { content: "\2605"; color: greenyellow; }
```

- Will prefix any anchor with a light green star

I AM DIV THAT CONTAINS SOME LONG NONSENSE SO THAT I SPAN multiple lines. The text was generated at ★<https://www.lipsum.com/>. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris in lacus mollis, commodo erat eget, gravida metus. Praesent ullamcorper eget libero sed molestie. Vestibulum quis eleifend leo. Aliquam pulvinar quam velit. Cras in elementum sem. Interdum et malesuada fames ac ante ipsum primis in faucibus. Duis varius efficitur purus, nec euismod ipsum molestie vitae. Pellentesque mauris augue, mattis quis mollis pellentesque, vehicula non diam.





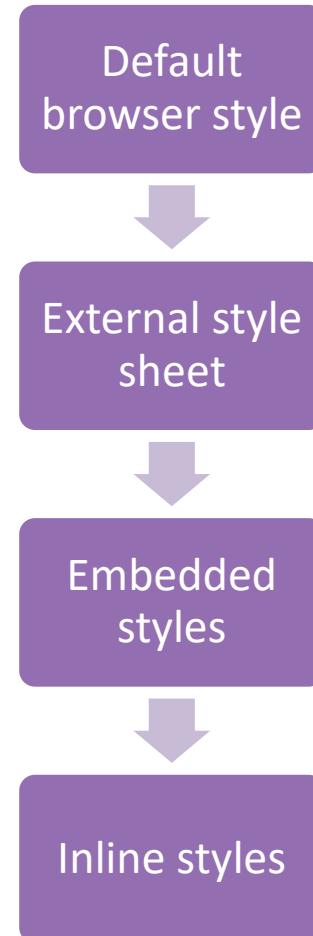
Conflicting rules





The cascade

- We can write styles inline, embedded in the page, and in separate sheets
 - What if they specify conflicting rules?
 - How do we decide which to apply?
- Answer: *cascade* rules
- The conflicting style rule that was defined last in the source order wins
 - i.e. the one defined ‘closest’ to the element wins





Specificity

- Multiple, conflicting rules might apply to a single element
 - E.g. a paragraph with a class name and an id
- So how do we decide which rules to apply?
- Answer: *specificity* rules
- The most specific rule wins





Inheritance

- Some CSS property values set on parent elements are inherited by their descendants
 - E.g. color and font styles
 - Can be overridden by specify styles for the descendants
- Some CSS property values are not inherited by their descendants
 - E.g. widths, margins, padding, and borders
- Which are inherited by default is designed to be intuitive
 - But consider what happens if you specify font size as a percentage...

great grand parent
grand parent
parent
child



Recommended Reading





More on HTML5...

- Get Ready for HTML5 (from 2009)
 - <http://www.alistapart.com/articles/get-ready-for-html-5/> Accessed August 2021
- John Allsopp is on the Fence
 - <http://www.alistapart.com/articles/semanticsinhtml5> Accessed August 2021
- HTML5 Developer Guide to learning HTML
 - <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML> Accessed August 2021





More on CSS...

- “Everything *you* need to know about CSS3”
 - <https://www.css3.info/>
- Eric Meyer on CSS
 - <https://meyerweb.com/eric/css/>
- A list Apart on CSS
 - <https://alistapart.com/blog/topic/css/>
- CSS Validation service at W3C
 - <https://jigsaw.w3.org/cssvalidator/>





CSS Reference Documentation

- CSS Properties reference
 - <https://www.w3schools.com/cssref/default.asp>
- CSS Selectors reference
 - https://www.w3schools.com/cssref/css_selectors.asp
 - <https://www.w3.org/TR/selectors-3/>
- CSS Units reference
 - https://www.w3schools.com/cssref/css_units.asp





CSS Tutorials

- Full CSS Tutorials
 - <https://www.w3schools.com/Css/>
 - <https://www.javatpoint.com/css-tutorial>
- CSS Beginner Tutorial
 - <https://www.htmldog.com/guides/css/beginner/>





More on Web Standards...

- Web Standards Project – “Our work here is done”
 - <http://www.webstandards.org/> Accessed August 2021
- Acid Tests – “the home of the Web Standards Compliance Acid Tests”
 - Controversial and not recommended as a test for standards compliance but interesting!
 - <http://www.acidtests.org/> Accessed August 2021
- Tim Berners Lee – “One Small Step for the Web...”
 - <https://www.inrupt.com/blog/one-small-step-for-the-web> Accessed August 2021





W3C and WHATWG to work together to advance the open Web platform

W3C AND WHATWG TO WORK TOGETHER TO ADVANCE THE OPEN WEB PLATFORM



I am pleased to announce that W3C and the WHATWG have just signed an agreement to collaborate on the development of a single version of the HTML and DOM specifications.

- <https://www.w3.org/blog/2019/05/w3c-and-whatwg-to-work-together-to-advance-the-open-web-platform/> Accessed August 2021





Conversational Semantics: HTML & ARIA

Conversational Semantics

by [Aaron Gustafson](#) · August 30, 2018

Published in [Accessibility](#), [HTML](#)

As Alexa, Cortana, Siri, and even customer support chat bots become the norm, we have to start carefully considering not only how our content looks but how it could sound. We can—and should—use HTML and ARIA to make our content structured, sensible, and most importantly, meaningful.

- <https://alistapart.com/article/conversational-semantics/> Accessed August 2021





Books

- *CSS3 Visual Quickstart guide* 6th Edition, Jason Cranford Teague, (Peachpit Press)
- *HTML5 and CSS3 Visual Quickstart guide*, Elizabeth Castro, Bruce Hyslop (Peachpit Press)
- *The ZEN of CSS Design: Visual Enlightenment for the Web*, Dave Shea, Molly E. Holzschlag (Peachpit Press)
 - Case studies based on 36 of the designs from <http://www.csszengarden.com/>





Required Reading & Consolidation





Reading

- Week 3 required reading on Student Resources
 - 4 articles
 - Pay particular attention to the dates they were written
 - Compare and contrast to what you have read and learned from other resources
 - <https://studres.cs.st-andrews.ac.uk/IS5103/Lectures/W03-HTML5-CSS/Reading>





Consolidation

- Week 3 consolidation exercise
 - HTML5 & CSS
- CSS Selectors Tutorial
 - https://www.w3schools.com/Css/css_selectors.asp
- CSS Selectors Exercises
 - <https://www.w3schools.com/Css/exercise.asp>
- CSS Quiz
 - https://www.w3schools.com/Css/css_quiz.asp





IS5103 Web Technologies: More CSS

Dr Ruth Letham





Overview

- CSS3: Benefits & challenges
- Multi-column Layouts
- Fonts & Text Effects
- Managing CSS
- Design & development





CSS3

Completed/Stable

- Fonts
 - @font face rule
- Media Queries
 - @media rule
- Backgrounds & Borders
 - Border images, gradients, radius, shadow
 - Multiple background images
- Multi-column layout
 - Within a single element
- Flexible Box layout
 - Flexible flow layout

Testing/Refining

- Grid Layout
 - 2-dimensional layout
- Speech
 - Rendering via speech synthesis
- Transforms
 - Scale, rotate, skew, move (in 2-D space)
- Cascading Variables
- Transitions
 - Smooth transform over time
- Animations
 - @keyframes rule





CSS3 Enhancements

- Benefits
 - Reduced need for JavaScript
 - › Reduced reliance on developers to produce design/display features
 - › Reduced download size
 - Reduced need for images
 - › Reduced download size
 - Download & display faster on less powerful devices
 - › vs. historical complicated/programmatic techniques
- Challenge
 - Cross-browsers compatibility and support!





CSS Browser/Vendor prefixes

- Problem: How to introduce new features but not break cross-browser support?
- Solution(?): Browser/Vendor prefixes
 - Prefix onto proposed new properties
 - Prefix onto non-standard properties
- Specify multiple targets for a style
 - One for each relevant prefix
 - Make sure last target is non-prefixed
- Problem: exact syntax may change

Prefix	Browser	Examples
-moz-	Firefox, Camino	-moz-tab-size
-ms-	IE, Edge (old)	-ms-hyphens
-o-	Opera	-o-tab-size
-webkit-	Chrome, Safari, Android, iOS, Edge	-webkit-initial-letter





Multi-column Layouts





In the beginning: Floats

- We have already seen *block* elements and the *box model*
 - Elements appear on their own line with padding and margins around the content
- To make block elements appear on the same line as each other you can *float* them
 - left or right, inherit from parent, none (display where it naturally occurs in the page)
- Width will be determined based on content
 - Need to set width if you want multiple elements on the same line regardless of content, e.g. ~33% for 3 elements



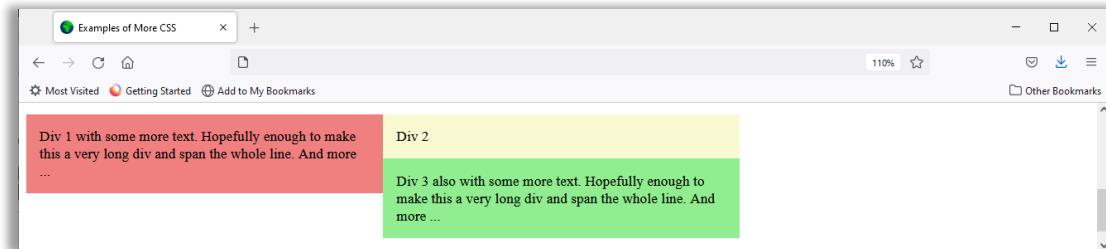


Problems with floats

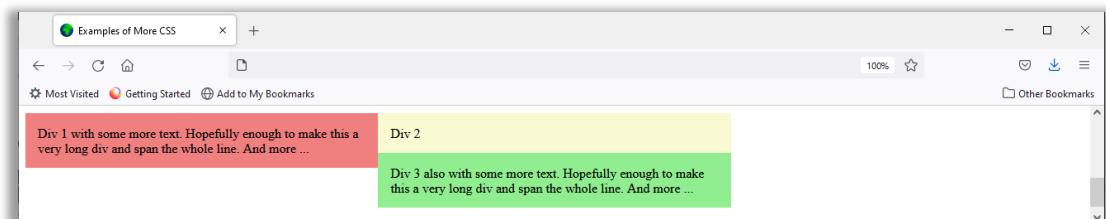
- Difficult to calculate widths
 - How to account for margins, borders & padding?
 - Browsers have different defaults
 - 33% often not 3 on a row!
- Fragile
 - Display changes with zoom
 - Display changes with page width
- Complex
 - Nested divs
 - Mixing block and float
 - Floated elements ‘overflowing’ their parent



31% width with 15px padding



Zoom changed to 110%



Zoom at 100%, browser width reduced by 2px





In the beginning: Tables

- We have already seen that tables have been used for layout
 - Avoids the problems of floats
- But not meaningful markup
 - Layout ≠ tabular data!!!
- Developers needed a better way to specify layouts
 - No longer just presenting documents and data
 - Need menus, side bars, headers, footers, columns...





Flexible layouts

- Enter the *flex* layouts
 - New display type
 - › Evolved from ‘box’ and ‘flexbox’
 - In addition to *block* & *inline*, we have *flex*
- Similar to float ... but with more control
 - Fixes *some* of the problems
- Specify that an element’s contents will have flexible layout
 - The direction they will stack: side-by-side or vertically
 - The way contents will be aligned within their rows
 - The way rows will be aligned in the container
- Specify how an element within a flex container will grow and shrink





Example

```
.flex-container {  
    display: flex;  
    flex-wrap: wrap;  
    justify-content: space-between;  
    align-items: stretch;  
}  
  
.flex-container > div {  
    flex-grow: 1;  
    flex-shrink: 1;  
    flex-basis: 44%;  
}
```

My Fake Blog

First news story of my fantastic blog! It is rather long because the blog was new and I was all excited about writing it! Read all about it!

Second news story of my fantastic blog! It is a bit shorter as enthusiasm wanes. Read all about it!

Third news story of my fantastic blog! Shorter still. Read all about it!

Fourth news story of my fantastic blog! Read all about it!

Fifth news story of my fantastic blog! Read all about it!

Sixth news story of my fantastic blog! Read all about it!

My Fake Blog

First news story of my fantastic blog! It is rather long because the blog was new and I was all excited about writing it! Read all about it!

Second news story of my fantastic blog! It is a bit shorter as enthusiasm wanes. Read all about it!

Third news story of my fantastic blog! Shorter still. Read all about it!

Fourth news story of my fantastic blog! Read all about it!

Fifth news story of my fantastic blog! Read all about it!

Sixth news story of my fantastic blog! Read all about it!

My Fake Blog

First news story of my fantastic blog! It is rather long because the blog was new and I was all excited about writing it! Read all about it!

Second news story of my fantastic blog! It is a bit shorter as enthusiasm wanes. Read all about it!

Third news story of my fantastic blog! Shorter still. Read all about it!

Fourth news story of my fantastic blog! Read all about it!

Fifth news story of my fantastic blog! Read all about it!

Sixth news story of my fantastic blog! Read all about it!





Multi-column layouts

- Flex is good when you have elements that you want to flow, grow and shrink inside some container
- But what if you just want some content that will naturally flow over multiple columns?
 - E.g. text for an article
- Enter *multi-column* layouts
 - Specify how many columns an element should have
- Based on the content and number of columns, browser works out the height and flows it across the columns





Example

```
.three-col {  
    column-count: 3;  
    column-rule: thick solid aquamarine;  
    column-gap: 60px;  
}
```

My Fake News Article

This is a news article that is fake. Not an attempt to proliferate fake news. Again, the rest of this is just filler text to make the article long enough to do something interesting!

This is a new paragraph with some long nonsense so that I

span multiple lines. The text was generated at <https://www.lipsum.com/>. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris in lacus mollis, commodo erat eget, gravida metus. Praesent ullamcorper eget libero sed molestie. Vestibulum quis eleifend leo.

This is a shorter nonsense paragraph. Aliquam pulvinar

My Fake News Article

This is a news article that is fake. Not an attempt to proliferate fake news. Again, the rest of this is just filler text to make the article long enough to do something interesting!

This is a new paragraph with some long nonsense so that I span multiple lines. The text was generated at <https://www.lipsum.com/>. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris in lacus mollis, commodo erat eget, gravida metus. Praesent ullamcorper eget libero sed molestie. Vestibulum quis eleifend leo.

This is a shorter nonsense paragraph. Aliquam pulvinar

<https://www.lipsum.com/>.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris in lacus mollis, commodo erat eget, gravida metus. Praesent ullamcorper eget libero sed molestie. Vestibulum quis eleifend leo.

This is a shorter nonsense paragraph. Aliquam pulvinar

in elementum sem.

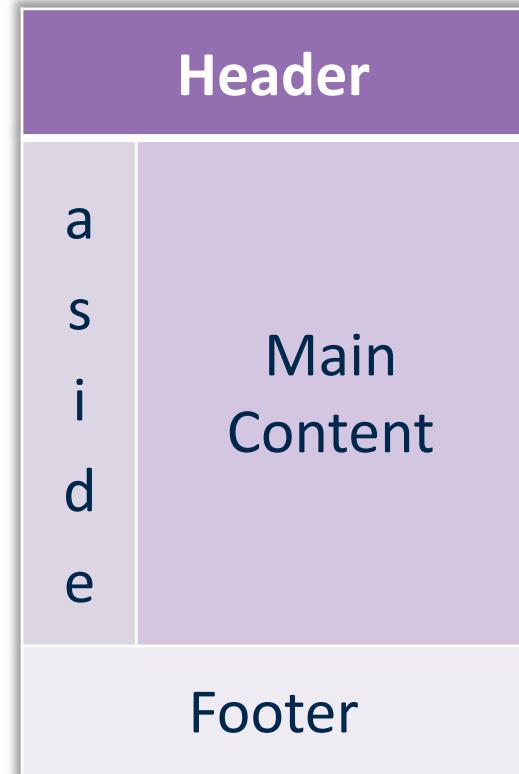
This is my concluding paragraph with just a bit more nonsense. Interdum et malesuada fames ac ante ipsum primis in faucibus. Duis varius efficitur purus, nec euismod ipsum molestie vitae. Pellentesque mauris augue, mattis quis mollis pellentesque, vehicula non diam.





2-D Layouts

- Flex and column-count are *flow* layouts
 - We can make output look like a grid but it is not 2-D
 - No concept of both rows and columns
- But what if you need to represent a 2-dimensional layout?
 - E.g. header, sidebar, content, footer?
- Enter *grid* layouts
 - Specify how contents are displayed in rows *and* columns





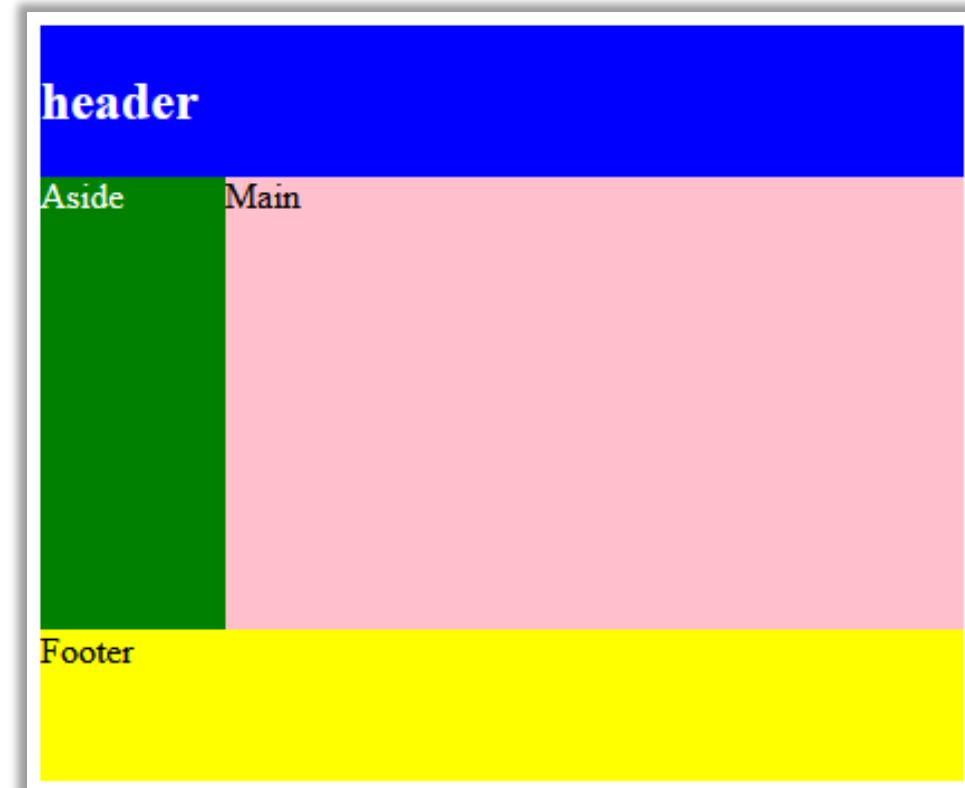
Example: row & column

- Specify template for rows & columns

```
#page {  
  display: grid;  
  grid-template-rows: 1fr 3fr 1fr;  
  grid-template-columns: 1fr 4fr;  
}
```

- The header and footer span from column 1 (inclusive) to column 3 (exclusive)

```
#page > header, #page > footer {  
  grid-column: 1 / 3;  
}
```





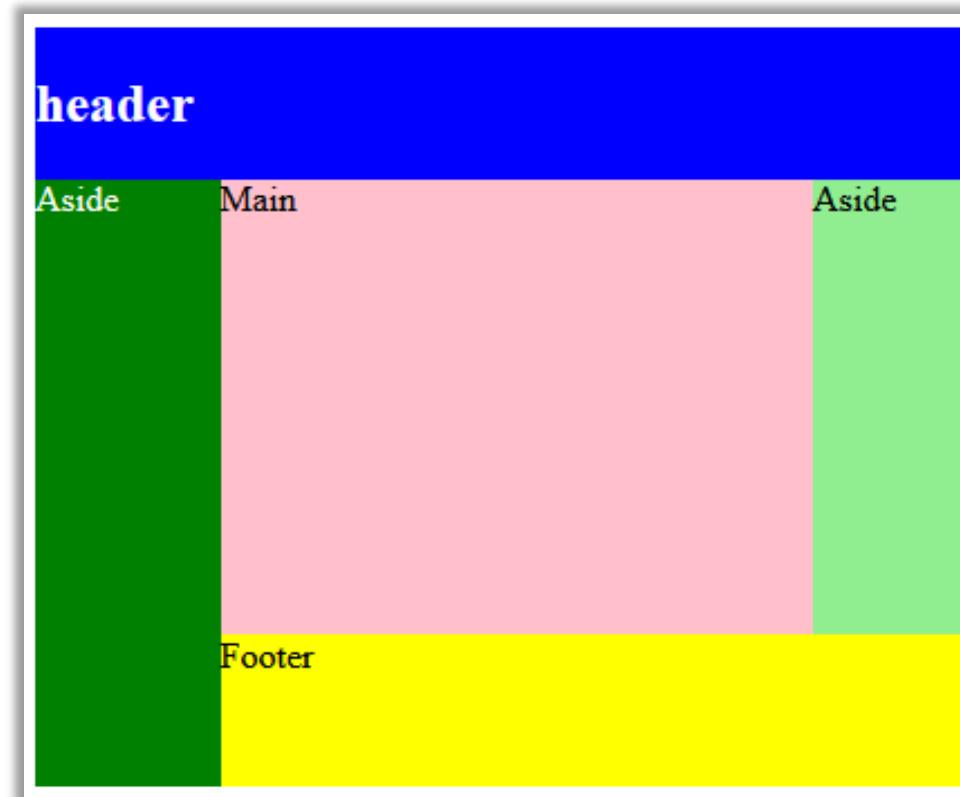
Example: area

- For more complex layouts , specify a template area

```
#page {  
  display: grid;  
  grid-auto-rows: 1fr 3fr 1fr;  
  grid-auto-columns: 3fr 12fr 2fr;  
  grid-template-areas: "hd hd hd"  
    "la mn ra" "la ft ft";  
}
```

- Associate contents with an area

```
#page > header { grid-area: hd; }  
#page > #la { grid-area: la; }  
#page > #ra { grid-area: ra; }  
#page > #main { grid-area: mn; }  
#page > footer { grid-area: ft; }
```





Fonts & Text





Font Family

- CSS property
 - Specify in the font to be used for a CSS target
- Specify a list of fonts
 - If the first cannot be applied by the browser, fallback to the next

- Examples:

font-family: Arial, Helvetica, sans-serif;

First choice Arial,

Fallback to Helvetica if necessary,

Last resort to the browsers default sans-serif font.





Web Fonts @font-face

- There are limited ‘web-safe’ font choices
 - These provide consistency across devices
 - Include generic families: serif, sans-serif, monospace, cursive, fantasy, MS fonts
- To increase options, you can associate a font file with your website
 - Use @font-face rule
 - Self hosted or using a font service
- Still need to use font-family to link your font to CSS targets
- Some fonts have commercial licences for use on the web
 - As with all sources, make sure you check licences before using!

<https://blog.hubspot.com/website/web-safe-html-css-fonts> Accessed Sept 2021





Font-face example

- Set up the locally hosted font and call it ‘alexbrush’

```
@font-face {  
    font-family: alexbrush;  
    src: url("fonts/alex-brush/AlexBrush-Regular.ttf");  
}
```

- Attach the font to paragraphs with the class ‘handwritten’

```
p.handwritten {  
    font-family: alexbrush, cursive;  
}
```

Sample of Alex-Brush font. Does it look hand-written?





Text Shadow

- Not a font, but related to text display
- Specify the horizontal offset, vertical offset, amount of blur, and colour of the shadow
- Examples:

`text-shadow: 2px 2px 5px #999;`

For a subtle, grey shadow

Text Shadow

`text-shadow: 12px 7px 1px #c73ec7;`

For an over-the-top pink shadow

Sample text with a shadow.
Sample text with a shadow.





Managing CSS





Comments

- Notes to human readers
 - Open with /* close with */
- Ignored by browsers & user agents
- Used for
 - Reminders, guidance, or instructions for designers
 - Organising styles into logically different parts
 - › E.g. icons, fonts, menus, sidebars,

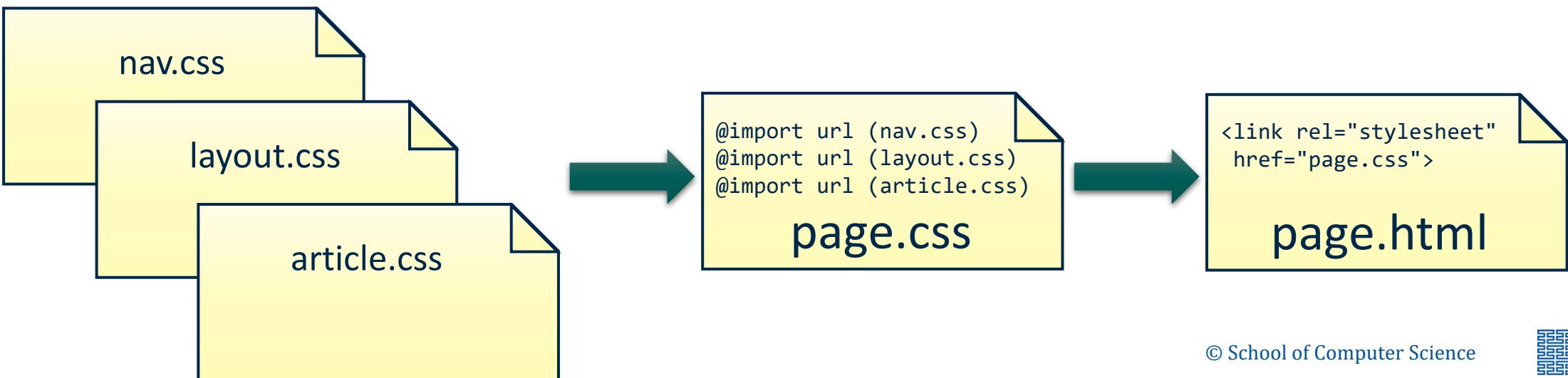
```
/*
COLOUR PALLET
* corflower blue (#6495ED)
  - border & text for aside
* yellow (#FFFF00)
  - background for highlighting
* medium violet red (#C210A0)
  - for banded text
*/
```





Separate CSS files

- Partition files for different logical elements
 - Need to do this depends on size of project
 - Overdo it and you have a maintenance headache
- Import (serial loading) Vs. Link (parallel loading)
 - <https://www.thoughtco.com/difference-between-import-and-link-3466404>
 - <http://www.stevesouders.com/blog/2009/04/09/>





Optimising CSS

- Reduce the amount of CSS you write
 - We do not have to style every element individually
 - › Inheritance, grouping, dependent selectors, classes
 - › But don't over use dependents!
 - Use the right layout
 - › Flex & grid require less code vs floats
- Avoid over-use of 'expensive' properties that require recalculation
 - Border-radius, box-shadow, opacity, transform, fixed position
- Use CSS transitions & animations instead of JavaScript equivalents





Minifying & Compressing CSS

- Reduce file size before deploying site
- Create a **minified** version: ~15% reduction in file size
 - Removes all the things that make it more human readable
 - E.g. comments, white space, etc.
- Create a **compressed** version: ~75% reduction in file size
 - Applies various algorithms (e.g. LZ77, Huffman coding, run-length-encoding, ...)
- Compress and minify: ~80% reduction in file size
- Experiment with options: TEST & **keep the readable version!**





Design & Develop

Test & Troubleshoot





CSS Development

- Use browsers' in-built web development tools
 - To inspect & modify your CSS
- Use online sandboxes e.g.
 - CodePen: <https://codepen.io/>
- Use CSS generation tools e.g.
 - The Ultimate CSS Generator: <https://webcode.tools/generators/css>
 - CSS Code Generators: <https://html-css-js.com/css/generator/>
 - CSS3 Maker: <https://www.toptal.com/developers/css3maker/>
 - › not actively maintained any more

The screenshot shows a browser window with developer tools open. The title bar says "Examples of More CSS". The main content area has a large bold header "Fonts, Text Effects & Layouts". Below it is a section titled "Fonts & Text Effects" with a sub-section "Sample text with a shadow". The developer tools interface includes tabs for Inspector, Console, Debugger, Network, Style Editor, and Performance. The Style Editor tab is active, showing the CSS code for the h1 element:

```
h1 {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 72px;
    line-height: 1em;
    text-shadow: 2px 2px 5px #999;
}
```

Below the browser window is a screenshot of the CodePen interface. It shows a pen titled "Untitled" with the name "Captain Anonymous". The preview area displays the same "Fonts, Text Effects & Layouts" page. The CodePen interface includes tabs for HTML, CSS, and JS, and a sidebar with "Console", "Assets", and "Shortcuts".





CSS Libraries & Frameworks

- Don't re-invent the wheel
 - Use ready built CSS libraries & frameworks
- Tried and tested solutions
 - Capable of handling range of design and functionality
- Loads to choose from...
 - W3.CSS: <https://www.w3schools.com/w3css/default.asp>
 - Skeleton: <http://getskeleton.com/>
 - Zurb Foundation: <https://get.foundation/>
 - Bootstrap: <https://getbootstrap.com/>
- Watch out for bloat
 - Only use what you *need*





Find Inspiration

- View the CSS code from other web sites by finding it in the page source
- Inspect the page CSS using a developer tools

```
<head>
  <meta http-equiv="Content-Script-Type" content="text/javascript" />
  <meta http-equiv="Content-Style-Type" content="text/css" />
  <link href="/mms/style/default.css" rel="stylesheet" type="text/css"></link>
  <link href="/mms/style/aqua.css" rel="stylesheet" type="text/css" title="Aqua"></link>
```

- But remember GAP





Debugging

- Not getting the results you hoped for?
- Check:
 - General typos in property/target/value names
 - Capitalisation typos
 - Using an = instead of a : between properties and values
 - Missing semicolons between rule
 - Quotes and braces are properly paired
- Use a process of elimination
 - Comment out sections
 - Use browser tools to inspect which rules affect the element
 - › Disable rules to see the effect





Debugging

- Check file extensions are correct
- Check files have uploaded
 - And are they in the right place?
- Check HTML as well as CSS
 - Is the link or @import correct?
- And remember, browsers sometime misbehave
 - Quit, empty cache, delete history, try another browser, ...





Testing: Browser Compatibility

- Try your page in multiple browsers
- Start with most standards compliant
 - Make it work
 - Take an incremental approach
- Add or investigate workarounds for troublesome browsers
 - Does the site still function without the ‘pretty’ features?
 - Are there browsers you won’t support?
- Use device simulators





Testing: Standards Validation

- Just like HTML, we can test standards compliance
 - <http://jigsaw.w3.org/css-validator/>

A screenshot of a Mozilla Firefox browser window showing the W3C CSS Validation Service. The title bar says "The W3C CSS Validation Service - Mozilla Firefox". The main content area has a blue header with the W3C logo and the text "CSS Validation Service" and "Check Cascading Style Sheets (CSS) and (X)HTML documents with style sheets". Below this are three tabs: "By URI" (selected), "By file upload", and "By direct input". A section titled "Validate by URI" asks "Enter the URI of a document (HTML with CSS or CSS only) you would like validated:" followed by a text input field labeled "Address:". Below the input field is a link "More Options".





Beyond CSS3





Extending CSS

- Syntactically Awesome StyleSheets (Sass) and Sassy CSS (SCSS)
 - Extension languages for CSS
 - **Not** an extension of the CSS standard
- Adds variables, maths operations, mixins and functions to CSS
 - Although CSS now has support for calculations & variables
 - › <https://www.w3.org/TR/css-values-3/>
 - › <https://www.w3.org/TR/css-variables-1/>
- SASS & SCSS are pre-compiled into CSS before publishing
 - Offer different syntaxes
 - › <https://sass-lang.com/documentation/syntax>
- Why Sass? By Dan Cederholm
 - <https://alistapart.com/article/why-sass>





Variables in Sass

Sass

```
$borders: thick solid #594F3B  
$margin: 8em  
$text-colour: white  
  
body  
  margin: $margin  
  color: $text-colour  
  
main  
  padding: $margin / 3  
  border: $borders
```

Generated CSS

```
body {  
  margin: 8em;  
  color: white;  
}  
  
main {  
  padding: 2.7em;  
  border: thick solid #594F3B;  
}
```





Variables in CSS

```
:root {  
    --borders: thick solid #594F3B;  
    --margin: 8em;  
    --text-colour: white;  
}  
  
body {  
    margin: var(--margin);  
    color: var(--text-colour);  
}  
  
main {  
    padding: calc(var(--margin)/3);  
    border: var(--borders);  
}
```



Declare the variables in the ‘root’ element so they can be used throughout the document

Use the variables in style definitions





CSS4?



ABOUT THE AUTHOR

Rachel Andrew is not only Editor in Chief of Smashing Magazine, but also a web developer, writer and speaker. She is the

MARCH 4, 2020 • [26 comments](#)

Why Are We Talking About CSS4?

QUICK SUMMARY Around the web and within the CSS Working Group, there has been some discussion about whether we should specify a version of CSS – perhaps naming it CSS4. In this article, Rachel Andrew rounds up some of the pros and cons of doing so, and asks for your feedback on the suggestion.

14 m
[CSS](#),
Share or [Link](#)

THERE HAS BEEN SOME DISCUSSION RECENTLY ABOUT

- <https://www.smashingmagazine.com/2020/03/css4-pros-cons-discussion/>





Recommended Reading





CSS3 Resources

- The Difference Between CSS2 and CSS3: Understanding the major changes to CSS3
 - <https://www.thoughtco.com/css2-vs-css3-3466978#css3-column-gaps-and-rules>
- CSS3 Specifications and drafts
 - <http://www.w3.org/Style/CSS/current-work>
- Demos of some CSS3 properties and rules
 - <http://www.css3.info/preview/>
- Guidance on CSS selector usage
 - <https://frontstuff.io/you-need-to-stop-targeting-tags-in-css>





Vendor prefixes

- Prefixes for different browsers
 - <https://www.thoughtco.com/css-vendor-prefixes-3466867>
- Do wee still need it?
 - <https://css-tricks.com/is-vendor-prefixing-dead/>





Layouts

- Flex boxes
 - <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>
 - https://www.w3schools.com/css/css3_flexbox.asp
- Grids
 - <https://css-tricks.com/snippets/css/complete-guide-grid/>
 - https://www.w3schools.com/css/css_grid.asp
- Multi-column
 - https://www.w3schools.com/css/css3_multiple_columns.asp





Layout Comparisons

- Grid vs. Flex
 - <https://blog.logrocket.com/flexbox-vs-css-grid/>
 - <https://webdesign.tutsplus.com/articles/flexbox-vs-css-grid-which-should-you-use--cms-30184>
 - https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout/Relationship_of_Grid_Layout
- Flex & Grid vs. multi-column layouts
 - <https://www.smashingmagazine.com/2019/01/css-multiple-column-layout-multicol/>





Font Sources

- Font Squirrel “Free Font Utopia”
 - <https://www.fontsquirrel.com/>
- MyFonts “professional fonts for any project”
 - <https://www.myfonts.com/>
- The League of Moveable Type “the open-source font foundry”
 - <https://www.theleagueofmoveabletype.com/>
- Adobe Fonts “Quality fonts at your fingertips”
 - <https://fonts.adobe.com/>
- Make sure you filter results for free web fonts
 - Still check licence files when you download fonts!





Font Resources

- Guide on how to choose your web font
 - <https://design.google/library/choosing-web-fonts-beginners-guide/>
- Web safe fonts
 - <https://blog.hubspot.com/website/web-safe-html-css-fonts>
- Web font licencing
 - <https://www.creativebloq.com/features/font-licensing>
- @font-face rule
 - <http://www.css3.info/preview/web-fonts-with-font-face/>
- A font designed for people with dyslexia (not free to use)
 - <https://www.dyslexiefont.com/>





Typography Resources

- Working with Typography
 - <https://learn.shayhowe.com/html-css/workingwith-typography/>
- Best practices for web typography
 - <https://typographyhandbook.com/>
- Variety of articles on typography (and web fonts)
 - <https://alistapart.com/blog/topic/typography-web-fonts/>





Optimising CSS Resources

- Minification Vs. Compression
 - <https://css-tricks.com/the-difference-between-minification-and-gzipping/>
 - <https://betterexplained.com/articles/how-to-optimize-your-site-with-gzip-compression/>
 - <https://blog.logrocket.com/the-complete-best-practices-for-minifying-css/>
- Tools
 - <https://csscompressor.com/>
 - <https://www.minifier.org/>
 - <https://cssminifier.com/>
 - <https://minifycode.com/css-minifier/>
- General tips on optimising CSS
 - <https://www.sitepoint.com/optimizing-css-performance/>
 - <https://frontstuff.io/you-need-to-stop-targeting-tags-in-css>





IS5103 Web Technologies: Planning

Dr Ruth Letham





Overview

- Research
- Design
 - Structure
 - Look & Feel
 - Content
- Testing
- Monitoring





Research





Purpose

- Clarify Aims & Objectives
 - What is the purpose of your site?
 - › sales, service, awareness, ... ?
 - Identify keywords
 - Define your Mission Statement
- Organise existing resources
 - Images, notes, literature, ...
- Identify & research your target market
 - Detailed demographics and psychographics
 - Social and behavioural data
 - Customer personas & story





Competitor Research

- Investigate similar sites
 - What do they do well? What could you do better?
 - Generate page topic ideas
 - Reverse engineer
- Competitive intelligence
 - Identify key competitors
 - Keyword analysis of sites
 - Analyse digital traffic
 - Generate “actionable insights” from market data





Competitive Insight Tools

- BuzzSumo “Run a search to quickly discover content ideas, uncover platform insights, identify passionate influencers and more.”
 - <https://buzzsumo.com>
- SimilarWeb “Understand your competitors' strategy and grow your market share.”
 - <https://www.similarweb.com>
- Semrush “Search Engine Results Page (SERP) tracker, site health checker, traffic analytics, backlink checker, ad analysis & more”
 - <https://www.semrush.com/>
- WhatRunsWhere “provides you with the digital ad intelligence you need to spend your dollars more efficiently and grow your business.”
 - <https://www.whatruswhere.com>





Design





“Taxing people's minds is counterproductive”

The Psychology of everyday things - Donald Norman

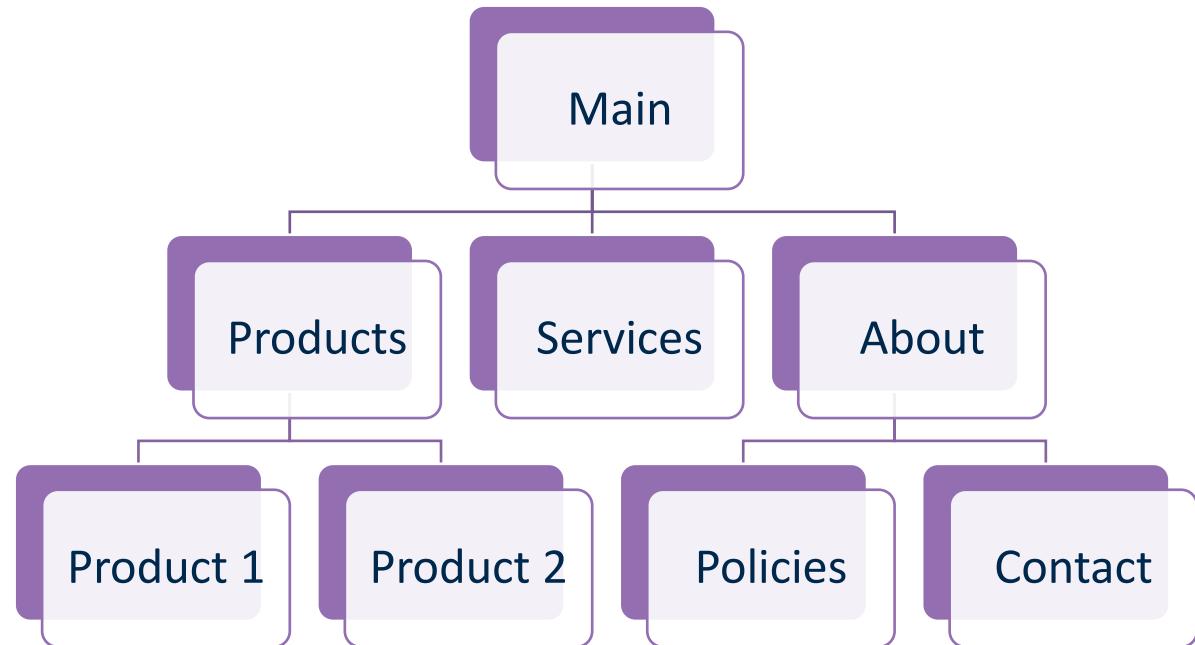
Memory, Language & Legibility





Site Plan

- Naming conventions
- Primary content
- Utility content
- Navigation
 - map ideas
 - user journey



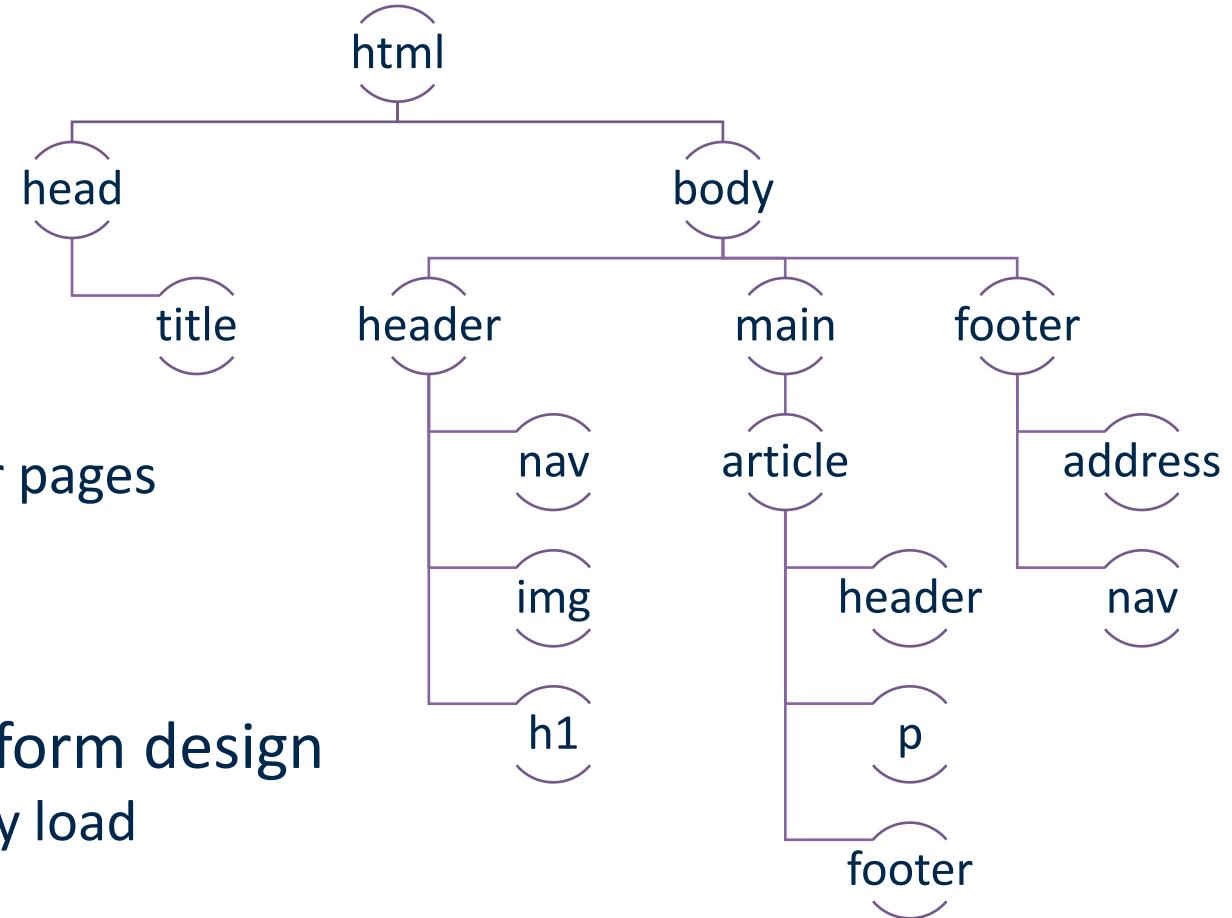
Section	Title	Keywords	Description
Index.html	Sunshine Tours	Sunshine Tours, Scotland, Uk, Holidays	Large Logo, Introduction, Top 10 destinations, Top 5 customer reviews





Page structure

- Outline major elements
 - Consistent use of mark-up
 - Navigational aids
- Identify common elements
 - What appears on all pages?
 - Element of home page on interior pages
 - Corporate identity
 - › Logo & corporate images
- Strive for consistency with a uniform design
 - Continuity & reduction of memory load





Colour Schemes

- Choose a colour scheme
 - Match your corporate image
 - Make it readable
 - Make it pretty(!)
- Coolors: <https://coolors.co/>
- Colorcombos: <https://www.colorcombos.com/>
- Colour lovers: <https://www.colourlovers.com/>
- Colors on the web: <http://www.colorsontheweb.com/>
- Colour scheme designer: <http://colorschemedesigner.com/>
- TPGi: Contrast analyser: <https://www.tpgi.com/color-contrast-checker/>





Typefaces

- Serif Vs. Sans Serif
- Designed for screen
 - Exaggerated x heights and larger typeface
 - Default OS, CSS font family Vs Web fonts

wide letter spacing

Good font

tall x-height

wide punch width

A close-up view of the words "Good font" in a dark blue sans-serif font. The letters are outlined with red dashed boxes to highlight specific typographic features: "Good" shows wide letter spacing between 'G' and 'o', and 'o' and 'd'; 'font' shows a tall x-height; and "font" shows a wide punch width for the 'f'. The background is white.

Verdana

Georgia





Legibility

- Add visual relief
 - Left hand margin
 - Contrast
 - White space
- Alignment of text
 - Fully justified? Left is even & predictable
- Text size
 - Relative size units in CSS: percentages (%) or proportional to base font size (em)
 - User preferences
 - Assistive technology





Line length & vertical spacing

- Eye's span of focus is three inches wide
 - Movement of head and eyes
 - Long trip back to left margin
- Vertical spacing
 - Generous leading
 - Compensate for line length
 - Lower resolution of screen





Prototype

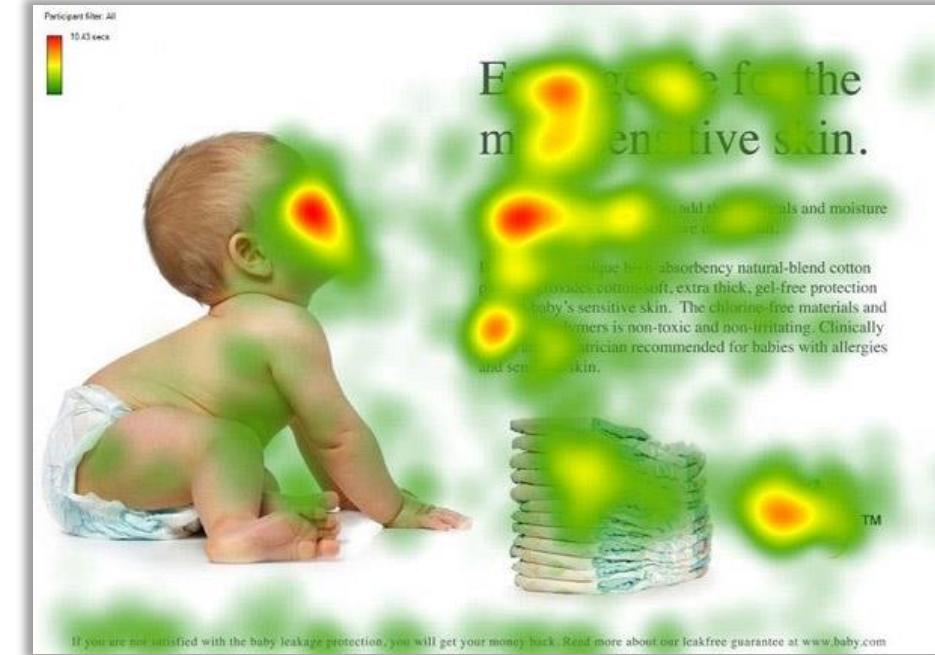
- Use prototyping tools to quickly create mock-ups
 - Site maps, wireframes, navigation
 - NOT content
- Wide choice of prototyping tools
 - Moqups: <https://moqups.com/>
 - Lucidchart: <https://www.lucidchart.com>
 - Vectr: <https://vectr.com/>
 - Cacoo: <https://cacoo.com>
 - HotGloo: <https://www.hotgloo.com/>
 - Mockingbird: <https://www.gomockingbird.com>
 - Invision: [https://www.invisionapp.com/](https://www.invisionapp.com)
 - MyBalsamiq: <https://balsamiq.com/>





Writing Content for the Web

- Need to know how users read on the web
- Scan page
 - Individual words or sentences
 - Images
- Passive vs active reading
- Use eye tracking to generate heat maps
 - What are readers looking at?



James Breeze as cited by
<https://www.throughlinegroup.com/2012/10/24/three-ways-to-control-the-audience-with-your-eyes/>





Eye Tracking



- F-pattern reading
 - <https://www.nngroup.com/articles/f-shaped-pattern-reading-web-content-discovered/> (2006)
 - <https://www.nngroup.com/articles/f-shaped-pattern-reading-web-content/> (2017)
- Key role in improving user experience
 - <https://www.usability.gov/get-involved/blog/2010/03/eyetracking.html>
- Marketing lessons can be drawn from eye tracking
 - <https://neilpatel.com/blog/eye-tracking-studies/>
- Watch a demonstration
 - https://www.youtube.com/watch?v=lo_a2cfBUGc





Scan read the web

- Make your text more scannable
 - Highlight keywords
 - Use *meaningful* subheadings
 - Use bullet lists
 - Express **one** idea per paragraph
 - Use the *inverted pyramid*
 - Reduce word count
- Guidelines not rules
- Need holistic approach
 - E.g. inverted pyramid & word count reduction does not work in isolation





Web Writing Tips

- Concise (word count)
- Headings (8)
- Sentences (20)
- Paragraphs (2-5 sentences)
- Pages (500)





Search results

I haven't entered your site yet....





Search Engine Optimisation (SEO)

- Search engines do not make their algorithms available
 - But google provides guidelines:
 - › <https://developers.google.com/search/docs/advanced/guidelines/how-search-works>
- Focus on actual content rather than meta data
 - Meaningful content & mark-up
- Improve ‘crawling’
 - Meaningful links & urls
 - Submit a sitemap or a crawl request for an URL
 - › <https://developers.google.com/search/docs/advanced/sitemaps/build-sitemap>
 - Maintain a robots.txt file to be read by a site crawler
- Increase connection with other sites





SEO & click conversion

- Need to convert high rankings in results to high click rate
- Page understood out of context
 - What does it offer?
- Relevancy of arrival point within site structure
- Short & Meaningful link navigation text
- Avoid mechanics of site/navigation
 - “Click here” has no semantic attachment





Testing

Functionality, accessibility, usability, compatibility & performance





Tools

- We have already seen tools to test your site during development
 - Browser Tools
 - Screen size testers like ScreenFly: <https://bluetree.ai/screenfly/>
 - W3C validators for HTML, CSS and links
 - ...
- ...but we should probably also test it on
 - Different browsers
 - Different connection speeds





Cross browser compatibility

- Could download browsers and open website in them
 - Chrome, Safari, Firefox, Edge, Opera, ...
 - ...but this would be time consuming and require manual analysis
- Better to use a tool e.g.
 - Cross-browser checkers like BrowserStack: <https://www.browserstack.com>
 - And web platform tests: <http://web-platform-tests.org/>
 - Automated checking of hundreds of browsers old and new
- Or pre-emptively check
 - Can I use: <https://caniuse.com/>





Performance optimisation

- Use tools to see how fast your website loads
 - Webpage Test: <https://www.webpagetest.org/>
 - › Can choose which browser to test against
 - › Also highlights potential security threats
 - Page Speed Insights: <https://developers.google.com/speed/pagespeed/insights/>
 - Pingdom Website Speed Test: <https://www.pingdom.com/>
- Follow best practice guides e.g.
 - Those you saw for CSS
 - More general guides like Yahoo's Best practice guide
 - › <https://developer.yahoo.com/performance/rules.html>





Regular test plan

- Decide:
 - What needs to be tested
 - When it need to be tested
 - How you are going to test it
- Record results

Title	File	Validation	Accessibility	File Size / Download Speed	Browsers	Links	Images
Sunshine Holidays	index.html	2019-20-12 PASS	WCAG FAIL	28k/3sec	2019-10-11 PASS	2019-10-11 PASS	2019-10-11 PASS

- Consider how testing corresponds to website updates and maintenance





Monitoring

Ongoing management & maintenance





Website Tracking

- What can you track?
 - Web server traffic
 - E-mail newsletter clicks
 - Banner ads (using website alias)
 - ...
- What do you hope to learn?
 - What are individuals looking at on your site/trends
 - Patterns of site usage
- Why is this important?
 - Refine and maximise marketing





Reputation Monitoring

- Just like testing, have a regular schedule and plan
 - Weekly routine
- Use monitoring tools e.g.
 - Trackur or SocialMention to monitor social media & study consumer opinions about brands
 - Google alerts to get alerts about keyword mentions on blogs, articles, press releases, etc.
 - Google analytics or Stat Counter to monitor how visitors engage with your site
 - TrustYou, Chatmeter, ReviewTrackers, or Reputology to gather and monitor reviews
 - SocialDraft to schedule content for digital marketing campaign





Business Performance

- Gain insights from statistics
 - New vs. repeat visitors
 - Conversion from visits to enquiries/purchases
 - Success of social media channels/campaigns

Week	% New Visitors	% Recurring visits	Landing Page	Exit Point	Enquiries submitted	Visitors from twitter	Visitors from facebook	Likes
2019-10-14	25	10	Product.html	Info.html	23	234	34	567





Recommended Reading





Writing for the web resources

- *Web Style Guide* by Patrick J. Lynch and Sarah Horton. Available at <http://webstyleguide.com/>
- *Don't Make Me Think, Revisited: A Common Sense Approach to Web (and Mobile) Usability* by Steve Krug (New Riders:2005)
- *Prioritizing Web Usability*, Jakob Nielsen Hoa Loranger (New Riders:2006)
- *CONTENT STRATEGY for the web*, Kristina Halvorson (New Riders:2010)
- Shorter articles
 - <https://www.nngroup.com/articles/how-users-read-on-the-web/>
 - <https://www.usability.gov/how-to-and-tools/methods/writing-for-the-web.html>
 - <https://www.uxbooth.com/articles/complete-beginners-guide-to-content-strategy/>

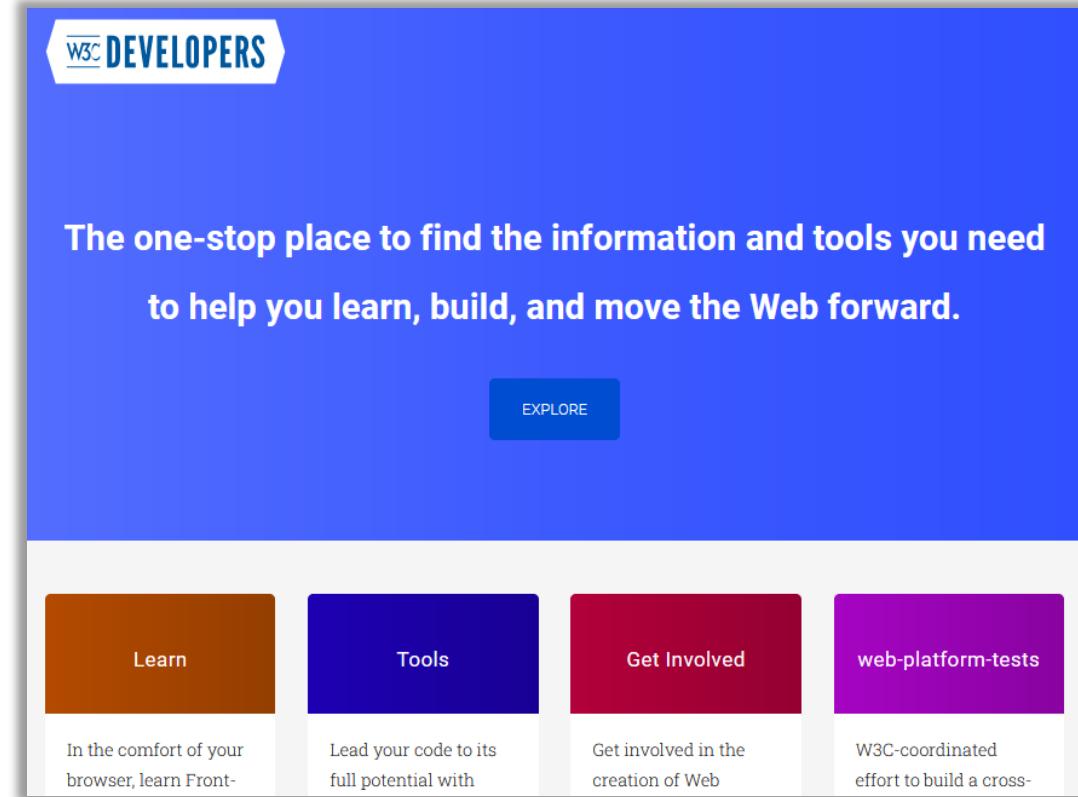




W3C Developers

“The one-stop place to find the information and tools you need to help you learn, build, and move the Web forward.”

<https://www.w3.org/developers/>



The screenshot shows the W3C Developers homepage. At the top, there's a blue header with the "W3C DEVELOPERS" logo and a tagline: "The one-stop place to find the information and tools you need to help you learn, build, and move the Web forward." Below the header is a large blue button labeled "EXPLORE". The main content area has four colored boxes: orange ("Learn"), dark blue ("Tools"), maroon ("Get Involved"), and purple ("web-platform-tests"). Each box contains a brief description: "In the comfort of your browser, learn Front-", "Lead your code to its full potential with", "Get involved in the creation of Web", and "W3C-coordinated effort to build a cross-", respectively.





Required Reading & Consolidation





Consolidation

- Week 4 consolidation exercises
 - Planning & CSS





Next...

- Accessibility





IS5103 Web Technologies: Accessibility

Dr Ruth Letham





“Accessibility is at the very core of standards-based design.”

Designing with Web standards Zeldman, Jeffrey, Marcotte, Ethan c2010





Overview

- What is web accessibility?
- Why is web accessibility important?
- Who is responsible for web accessibility?
- Overview of standards & guidance
- Conformance





What is web accessibility?

- The capability of a web page or web site to be viewed and used by **everyone**
- Often focusing on the needs of those with disabilities
 - But also need to consider those with any impairment (temporary or permanent)
 - And those with limited technologies (devices or connection)
- More specifically: people with impairments, disabilities, or limited technology can perceive, understand, navigate, interact and contribute
- Need to provide flexibility for different situations





Holistic view

- Technology
- Language
- Literacy
- Age
- Experience
- Device capability
- Affordability
- Developing world





Why is web accessibility important?

- It is at the heart of the world wide web

“The power of the web is in its universality.”

– Tim Berners Lee

- Grand ideals

– ...but also good for business

- Consider search engines crawling the web

– Examining mark-up and content

– No real sense of ‘visuals’

– The ultimate blind user: “The blind billionaire”





Assistive technologies

Input

- Touch Screen
- Voice input
- Modified Keyboard
- Head Wand or Mouth stick
- Sip and Puff device

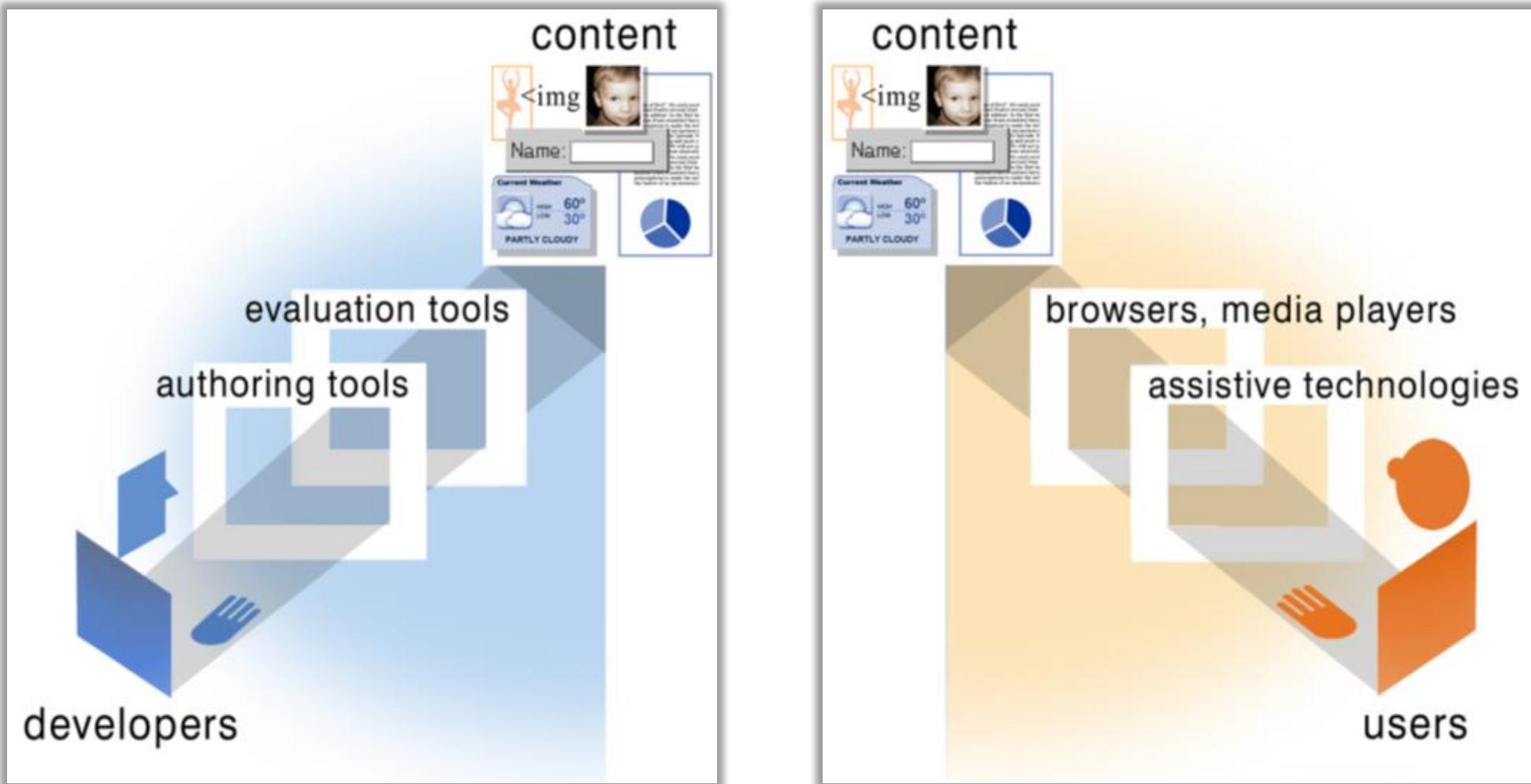
Output

- Screen magnifier
- Colour adjust
- Screen reader
- Braille device
- Text-only browsers





Multi-stakeholder development





Web Accessibility Initiative

- The Web Accessibility Initiative (WAI) is part of the W3C
- Address accessibility issues by developing
 - Guidelines for browsers, authoring tools and content creation
 - Educational resources
 - Validation tools
 - International accessibility standards
- Standards are created following W3C processes
 - <https://www.w3.org/WAI/standards-guidelines/w3c-process/>





Standards

- Web Content Accessibility Guidelines (WCAG)
- Authoring Tools Accessibility Guidelines (ATAG)
- User Agent Accessibility Guidelines (UAAG)
- Accessible Rich Internet Applications (WAI ARIA)
- Accessibility Conformance Testing (ACT)
- Evaluation and Report Language (EARL)
- Cognitive Accessibility
 - Informative guidance beyond the standards

<https://www.w3.org/WAI/standards-guidelines/> Accessed September 2021





WCAG guidelines are widely regarded as the **voluntary international standard** for web accessibility





WCAG Layers of Guidance

- WCAG 1.0 (<https://www.w3.org/TR/WCAG10>)
 - 14 guidelines, each with prioritised *checkpoints*
 - Checkpoints relate to conformance levels (A, AA, AAA)
 - Provide guidance on specific techniques in CSS & HTML to conform to each checkpoint
- WCAG 2.0 (<https://www.w3.org/TR/WCAG20>)
 - Added **4 principles** and replaced checkpoints with **success criteria**
 - Provided quick reference, understanding and techniques documents
- WCAG 2.1 (<https://www.w3.org/TR/WCAG21>)
 - Became current recommendation in June 2018
- WCAG 2.2 (<https://www.w3.org/TR/WCAG22>) currently a working draft





4 Principles

Perceivable

- Information being presented must not be invisible to all of a user's senses

Operable

- Interface must not require interaction that a user cannot perform

Understandable

- Content and operation must not be beyond a user's understanding

Robust

- Content must remain accessible as assistive technologies & user agents evolve





12 Guidelines

Perceivable

- Text Alternatives
- Time Based Media
- Adaptable
- Distinguishable

Operable

- Keyboard Accessible
- Enough Time
- Seizures and Physical Reactions
- Navigable
- Input Modalities

Understandable

- Readable
- Predictable
- Input Assistance

Robust

- Compatible

<https://www.w3.org/TR/WCAG21/> Accessed September 2021





WCAG 2.1 Quick Reference

- How to meet WCAG
 - <https://www.w3.org/WAI/WCAG21/quickref>
- Browse principles, guidelines & success criteria
- Filter by tags, conformance levels, technique categories and technologies.

How to Meet WCAG (Quick Reference)

A customizable quick reference to Web Content Accessibility Guidelines (WCAG)

Show About & How to Use

Contents Filter Hide

Selected Filters: WCAG 2.1: all success criteria and all techniques

WCAG Version: WCAG 2.1
Note: Clear Filters will not change the selected version.

Tags: Clear tags

- Developing
- Interaction Design
- Content Creation
- Visual Design

animation audio
auto complete autoplay
blinking buttons captcha
captions carousels

Show ALL TAGS

Levels: Select all

- Level A
- Level AA
- Level AAA

Techniques: Select all

- Sufficient Techniques
- Advisory Techniques
- Failures

Technologies: Select all

- HTML
- CSS
- ARIA
- Client-side Scripting

Principle 1 – Perceivable

Information and user interface components must be pres...

Guideline 1.1 – Text Alternatives

Provide text alternatives for any non-text content so that i...

1.1.1 Non-text Content — Level A

All non-text content that is presented to the user must have listed below. Show full description

Show techniques and failures for 1.1.1

Guideline 1.2 – Time-based Media

Provide alternatives for time-based media.

1.2.1 Audio-only and Video-only (Prerecorded)

For prerecorded audio-only and prerecorded video media, an alternative for text and is clearly labeled as such.

Show techniques and failures for 1.2.1

1.2.2 Captions (Prerecorded) — Level A

Captions are provided for all prerecorded audio and video media and is clearly labeled as such.

Show techniques and failures for 1.2.2





Success Criteria

- Techniques for each success criteria
- Divided into sufficient & advisory
- Includes failure criteria
- More detailed information available

1.4.1 Use of Color — Level A

Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or dis

Note 1: This success criterion addresses color perception specifically. Other forms of perception are covered in Guideline 1.3 including pro

Hide techniques and failures for 1.4.1 Sufficient Advisory Failures

Sufficient Techniques

Note: Other techniques may also be sufficient if they meet the success criterion. See [Understanding Techniques](#).

Situation A: If the color of particular words, backgrounds, or other content is used to indicate information:

- G14: Ensuring that information conveyed by color differences is also available in text
- G205: Including a text cue for colored form control labels
- G182: Ensuring that additional visual cues are available when text color differences are used to convey informa
- G183: Using a contrast ratio of 3:1 with surrounding text and providing additional visual cues on focus for links

Situation B: If color is used within an image to convey information:

- G111: Using color and pattern
- G14: Ensuring that information conveyed by color differences is also available in text

Advisory Techniques

- C15: Using CSS to change the presentation of a user interface component when it receives focus

Failures

- F15: Failure of Success Criterion 1.1.1 and 1.4.1 due to having a text alternative that does not include informatio
- F73: Failure of Success Criterion 1.4.1 due to creating links that are not visually evident without color vision
- F81: Failure of Success Criterion 1.4.1 due to identifying required or error fields using color differences only

<https://www.w3.org/WAI/WCAG21/quickref/?showtechniques=141#use-of-color> Accessed September 2021





Understanding Success Criteria

- More detailed than the quick reference
- Intent
- Benefits
- Examples
- Resources
- Techniques

Contents | GL: Distinguishable | Previous SC: Identify Purpose | Next SC: Audio Control

Understanding Success Criterion 1.4.1: Use of Color

Success Criterion [1.4.1 Use of Color](#) (Level A): Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.

This success criterion addresses color perception specifically. Other forms of perception are covered in [Guideline 1.3](#) including programmatic access to color and other visual presentation coding.

Intent

The intent of this Success Criterion is to ensure that all users can access information that is conveyed by color differences, that is, by the use of color where each color has a meaning assigned to it. If the information is conveyed through color differences in an image (or other non-text format), the color may not be seen by users with color deficiencies. In this case, providing the information conveyed with color through another visual means ensures users who cannot see color can still perceive the information.

<https://www.w3.org/WAI/WCAG21/Understanding/use-of-color> Accessed September 2021

On this page:

- Intent
- Benefits
- Examples
- Related Resources
- Techniques





Success Criteria Example: An examination

- Scenario
 - Students view an SVG image of a chemical compound and identify the chemical elements present based on the colors and numbers used in the diagram
- Accessibility
 - The text alternatives associated with each element name the color of the element and indicate the element's position in the diagram
- Result
 - Students who cannot perceive color have the same information about the compound as their classmates

<https://www.w3.org/WAI/WCAG21/Understanding/use-of-color#examples> Accessed September 2021





Techniques: Procedures

- Ensuring that information conveyed by color differences is also available in text
 1. Check that the information conveyed is also available in text and that the text is not conditional content
 - <https://www.w3.org/WAI/WCAG21/Techniques/general/G14>
- Using null alt text and no title attribute on img elements for images that AT should ignore
 1. Check that title attribute is either absent or empty.
 2. Check that alt attribute is present and empty.
 - <https://www.w3.org/WAI/WCAG21/Techniques/html/H67.html>





Common accessibility problems

- Video, audio, images and colour used to convey meaning
- Text that is poorly worded/structured
- Navigation & Hyperlinks that are not meaningful
- Forms with poor navigation, description, and error messages
- Tables with poor internal structure





Content & design

- Plan Heading Structure Early
- Ensure Logical Reading Order
- Provide Good Contrast
- Use True Text instead of Images of Text
- Use Adequate Font Size
- Remember Line Length
- Make Sure Links are Recognizable
- Design Keyboard Focus Indicators
- Design a "Skip to Main Content" Link
- Ensure Link Text Makes Sense on Its Own
- Design Usable Widgets and Controls
- Use Animation, Video, & Audio Carefully
- Do not Convey Content Using Only Color
- Design Accessible Form Controls



<https://webaim.org/resources/designers/#text> Accessed September 2021





Web Accessibility & General Myths

- ✖ Must design for the lowest common denominator
- ✖ Designs must be dull and text-only
- ✖ The web is inherently a graphical/visual medium
- ✖ Sites should be viewed the way the designer intended
- ✖ Building in accessibility takes too long
- ✖ If accessibility is not specified as a requirement, don't bother with it





Resources & Reading





Accessibility Validation

- WebAim
 - WAVE Web Accessibility Evaluation Tool <http://wave.webaim.org/>
 - Testing Web Content for Accessibility Quick Reference
<https://webaim.org/resources/evalquickref/>
- Taw web accessibility Test
 - <https://www.tawdis.net/>
- Cynthia Says: Free WCAG 2.0 & Section 508 Web Accessibility Scans
 - <http://www.cynthiasays.com/>
- Top 25 Awesome Accessibility Testing Tools for Websites
 - <https://dynamapper.com/blog/27-accessibility-testing/246-top-25-awesome-accessibility-testing-tools-for-websites>





Assistive Technologies

- 12 Best text-only browsers for browsing in slow internet connections
 - <https://thegeekpage.com/12-text-only-browsers-for-browsing-in-slow-internet-connections/>
- WebAim recommended screen readers
 - <https://webaim.org/articles/nvda/> (Windows)
 - <https://webaim.org/articles/jaws/> (Windows)
 - <https://webaim.org/articles/voiceover/> (MacOS)





Simulators

- PEAT Photosensitive Epilepsy Analysis Tool
 - <https://trace.umd.edu/peat>
- Colour blindness simulators
 - <http://www.color-blindness.com/coblis-color-blindness-simulator/>
 - <https://www.toptal.com/designers/colorfilter/>
- Inclusive Design Toolkit: Visual and audio impairment simulator
 - <http://www.inclusivedesigntoolkit.com/simsoftware/simsoftware.html>





Books

- Don't Make Me Think A Common Sense Approach to Web Usability, Steve Krug (New Riders:2005)
- Building Accessible Web Sites, Joe Clark (New Riders:2002)
- Prioritising Web Usability, Jakob Nielsen (New Riders:2006)
- Web Accessibility: Web standards and regulatory compliance, Thatcher (Springer 2006)





Reading on student resources

- Colour Blind Essentials
- WebAim's POUR quick Reference
- WebAim's WCAG Checklist
- Facing an Age-Old Problem
 - <http://dx.doi.org/10.1145/1562164.1562173>
- Working With Older Adults
 - <http://dx.doi.org/10.1016/B978-0-12-804467-4.00010-4>
- Dundee User Centre – A Space Where Older People & Technology Meet
 - <https://lemosandcrane.co.uk/resources/University%20of%20Dundee%20-%20Dundee%20User%20Centre%20-%20A%20space%20where%20older%20people%20and%20technology%20meet.pdf>





W3C Course

- Introduction to Web Accessibility
 - <https://www.edx.org/course/web-accessibility-introduction>





Typography & Readability

- 5 Keys to Accessible Web Typography
 - <https://betterwebtype.com/articles/2019/06/16/5-keys-to-accessible-web-typography/>
- Readability checkers
 - <http://juicystudio.com/services/readability.php>
 - <https://www.webpagefx.com/tools/read-able/>

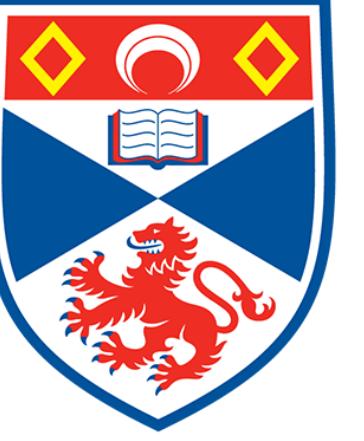




Next...

- This week
 - Choose your assignment topic & make a plan
- Week 6 (Independent Learning Week)
 - Work on W05 exercises
 - Work on your assignment
 - No scheduled classes
- Week 7
 - More on accessibility





University of
St Andrews

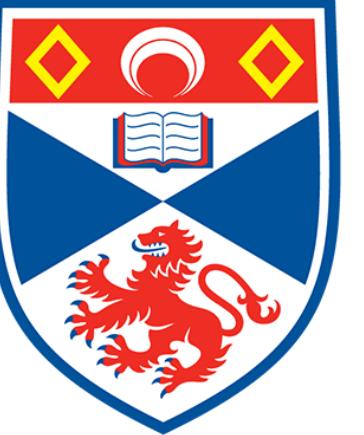
Responsive Web

Nnamdi Ekwe-Ekwe



University of
St Andrews

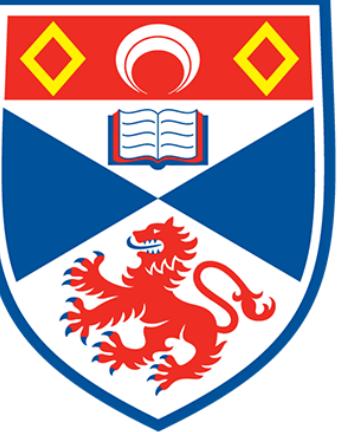
Responsive Web



University of
St Andrews

The state of devices

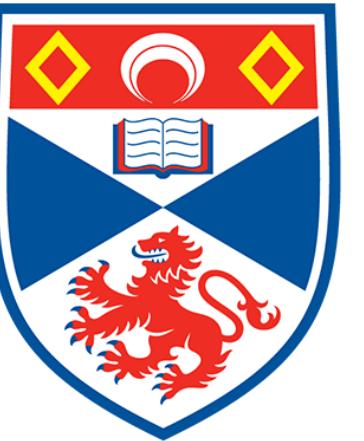
- We have a wide variety of devices being used today
- These are ever increasing in not only **number** but also in **variety**
- We can access the internet through our watch, tablet, smartphone, computer, etc.
- These devices are diverse with various characteristics that need to be accommodated for when building websites
- Remember our accessibility lectures!



University of
St Andrews

Two key questions

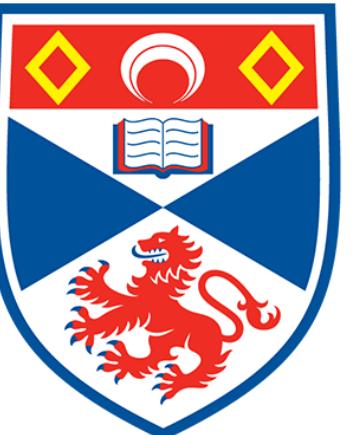
- How does one ensure that their website can be viewed the same across all these different devices?
- How can we deliver the same functionality to multiple web browsers/environments without unnecessary complexity?



University of
St Andrews

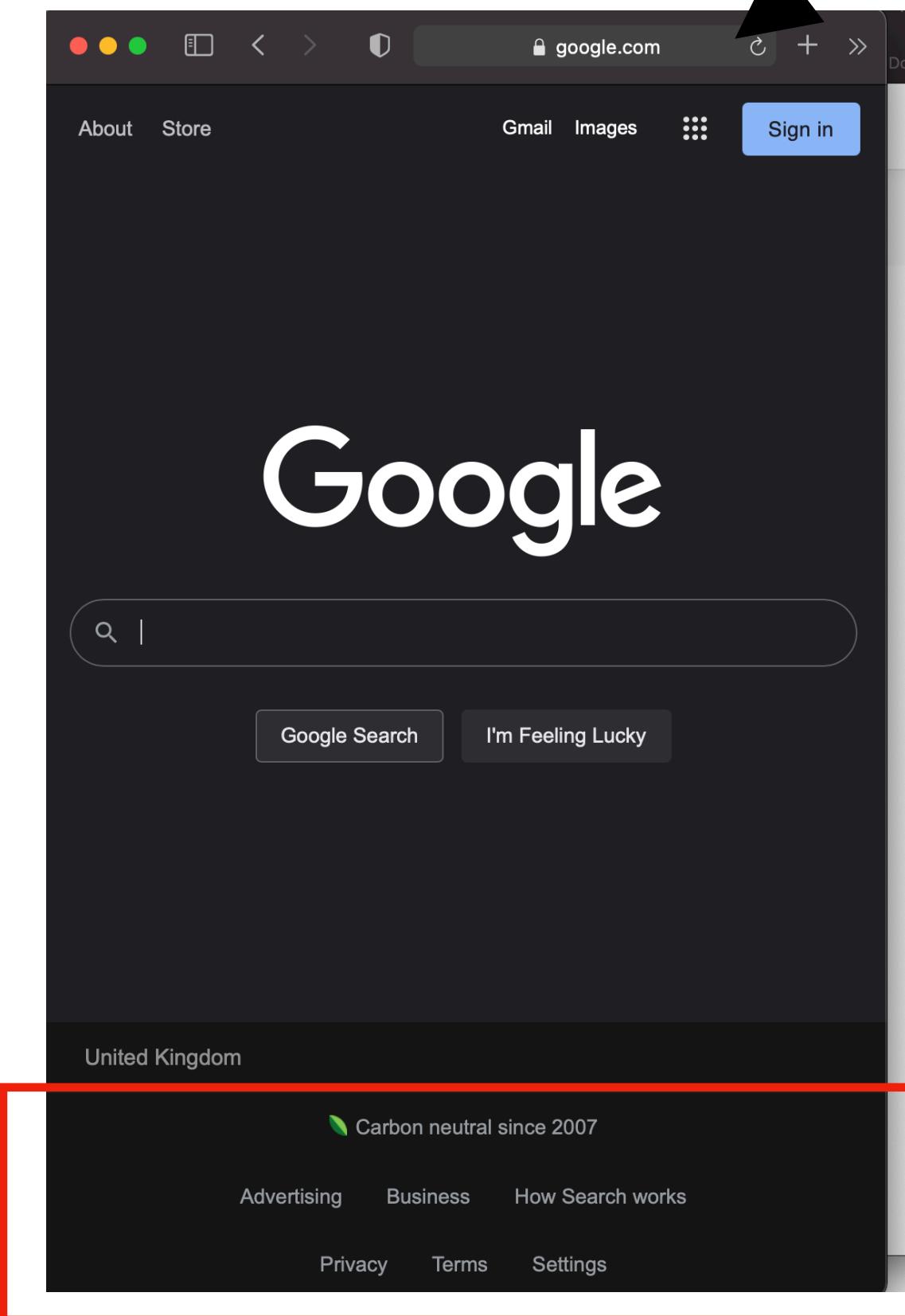
Responsive Web Design (RWD)

- “*Responsive web design (RWD) is a web development approach that creates dynamic changes to the appearance of a website, depending on the screen size and orientation of the device being used to view it.*”
- Source (<https://www.nngroup.com/articles/responsive-web-design-definition/>)
- “Write once, deploy anywhere”
- The developer just writes one website which then **responds** accordingly depending on the device it is viewed on
- RWD works based on screen size (determined by the number of pixels) and orientation (landscape or portrait)



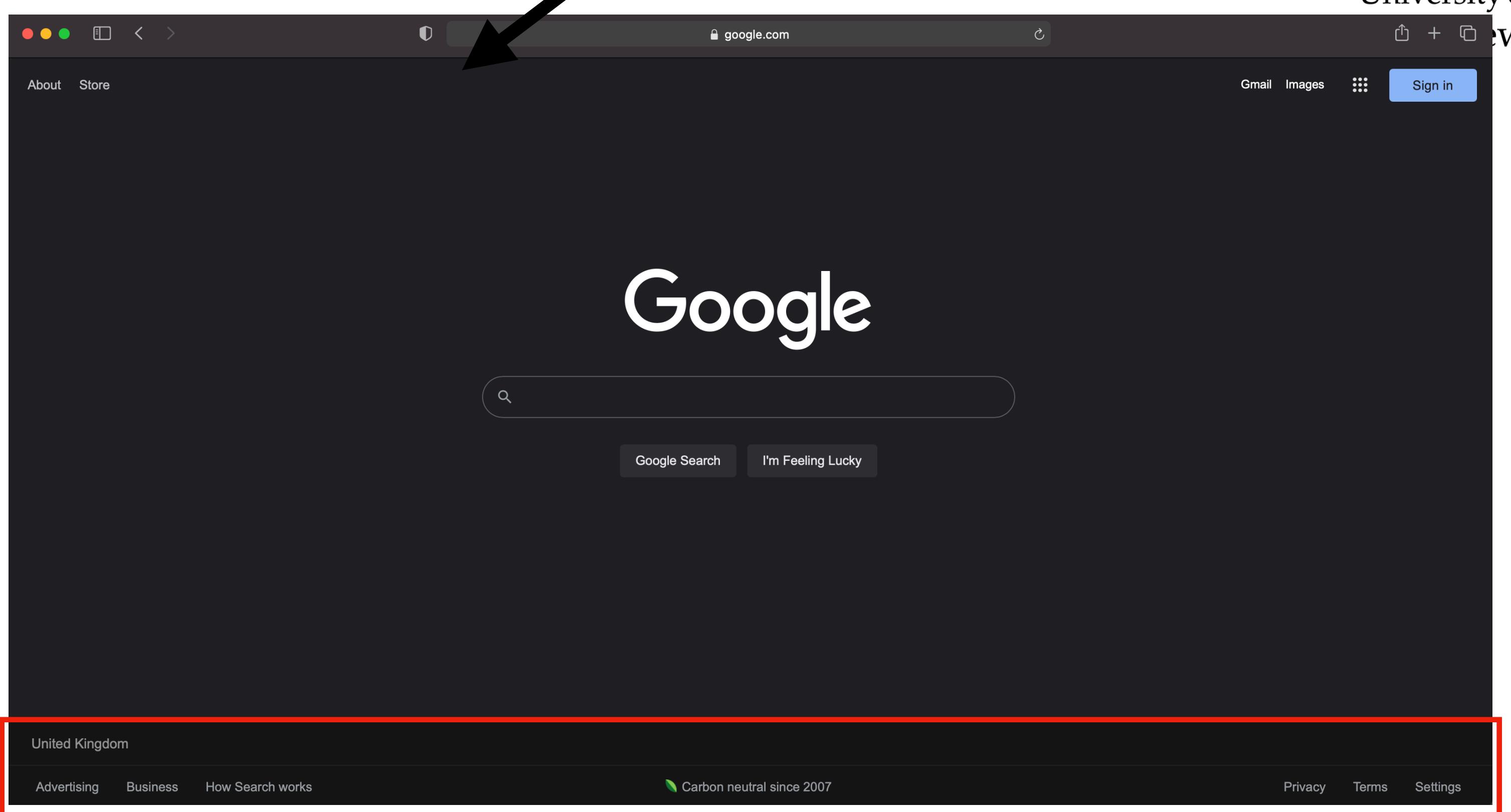
University of
St Andrews

Mobile

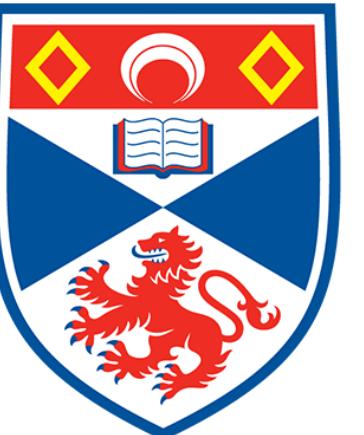


A screenshot of a Google search results page on a mobile device. The URL in the address bar is "google.com". The page shows the "About" and "Store" sections. A red box highlights the footer area, which includes links for "Advertising", "Business", "How Search works", "Privacy", "Terms", and "Settings". Below these links is a message: "Carbon neutral since 2007".

Desktop



A screenshot of a Google search results page on a desktop device. The URL in the address bar is "google.com". The page shows the "About" and "Store" sections. A red box highlights the footer area, which includes links for "Advertising", "Business", "How Search works", "Privacy", "Terms", and "Settings". Below these links is a message: "Carbon neutral since 2007".



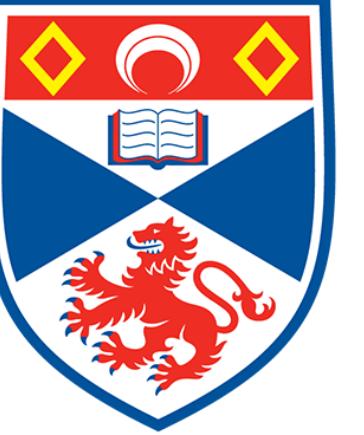
University of
St Andrews

Mobile

This screenshot shows the YouTube mobile website in dark mode. At the top, there's a navigation bar with a back arrow, forward arrow, and search icon. Below it is a header with the YouTube logo and a 'SIGN IN' button. A sidebar on the left contains links for Home, Explore, Subscriptions, Library, and History. The main content area features a 'YouTube Music' section with a 'GET IT NOW' button. Below this are video thumbnails for 'When Your Clap Game Is On Another Level' and 'I tried the BIGGEST Full English Breakfast'. A vertical sidebar on the right lists categories: Music, Sport, Gaming, Movies & shows, News, Live, Fashion & beauty, and Learning.

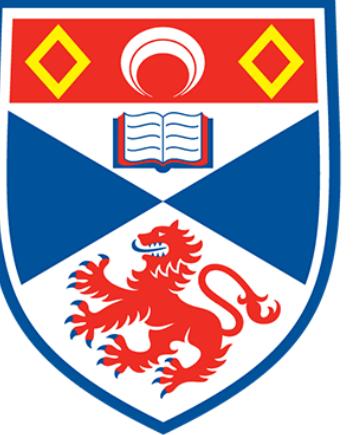
Desktop

This screenshot shows the YouTube desktop website in light mode. The layout is similar to the mobile version, with a top navigation bar, a header with the YouTube logo and 'SIGN IN' button, and a sidebar on the left. The main content area displays the same 'YouTube Music' section and video thumbnails for the same two videos. The vertical sidebar on the right also lists the same categories as the mobile version.



University of
St Andrews

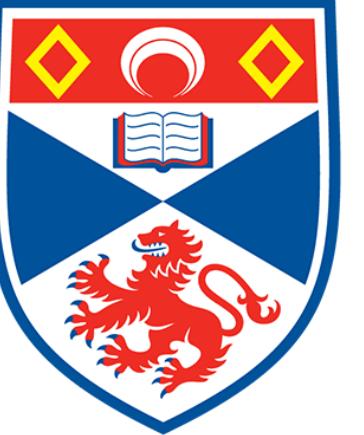
The Viewport



University of
St Andrews

How exactly is this accomplished?

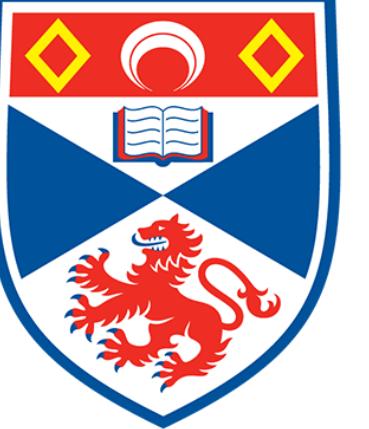
- It all begins with the **viewport**
- The viewport effectively denotes the user's visible area on a screen
- The size of the viewport is directly linked to the device's screen size
- **Smaller device = Smaller viewport; Larger device = Larger viewport**
- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`
- This goes into your main HTML page
- This line tells the website to fit/fill the width of the screen and initial-scale sets the initial zoom of the website when it is first loaded



University of
St Andrews

How exactly is this accomplished?

- Without this meta viewport tag, you may see content “run over” if it is greater than the width of the screen
- This can lead to users having to scroll vertically in order to try and see the rest of the content, rather than horizontally as normal



University of
St Andrews

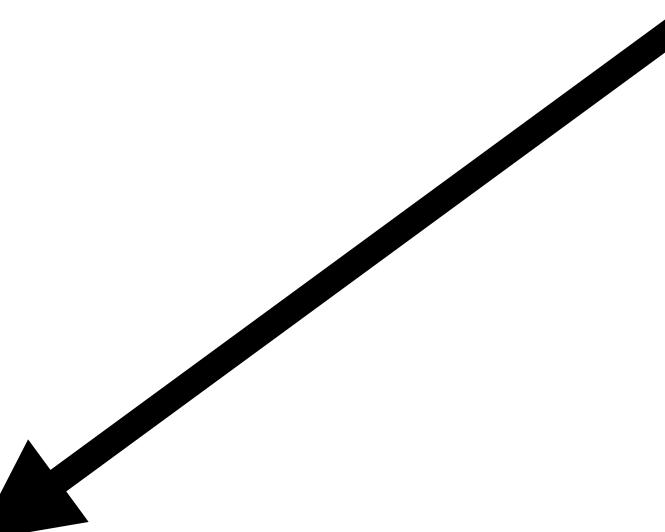
No viewport

To understand this example, you should open this page on a phone or a tablet.

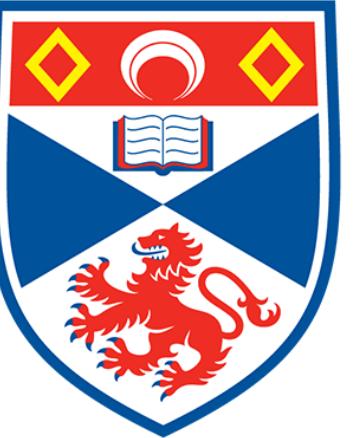


Picture runs over off the viewport - needs scrolling

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait



Source: https://www.w3schools.com/css/example_withoutviewport.htm



University of
St Andrews

Viewport

No running over

To understand this example, you should open this page on a phone or a tablet.



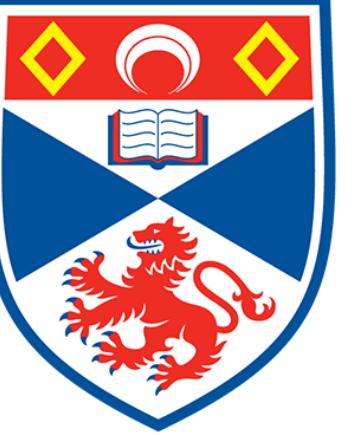
Content is contained within the screen viewport

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exercitation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option

Source: https://www.w3schools.com/css/example_withviewport.htm

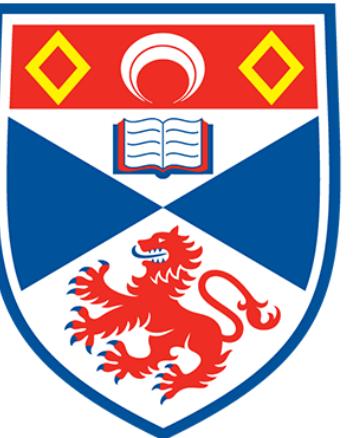
How can we make further use of these viewports?

- We can use the different viewports of devices to adapt our content based on those devices' screen widths
- Our website will **respond** to the various device viewports
- It may deliver different content or the content will be arranged differently on the page depending on the viewports
- We can accomplish this via the use of **CSS media queries**



University of
St Andrews

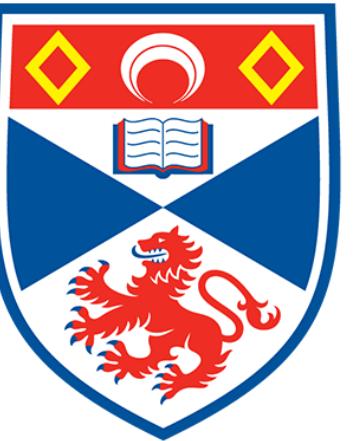
Media Queries



University of
St Andrews

Media Queries

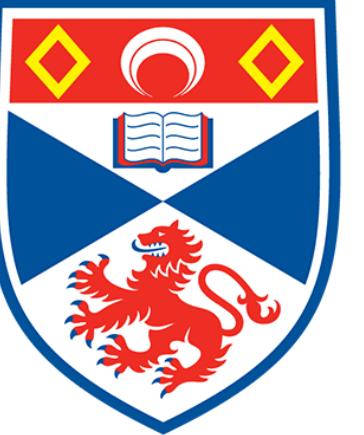
- Allow us to modify our website based on the device type as well as specific characteristics of the device
- For example, we can target screen devices or print devices (determining how our website will look on a screen or when printed out)
- We can also target different widths of a device (tablet, phone, computer, etc.)
- Using CSS, we can write various queries to accomplish our desired functionality



University of
St Andrews

Media Queries

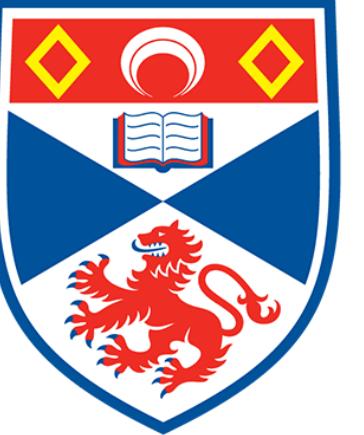
- Every media query consists of:
- Media types: **all**, **screen** (content viewed on a screen), **print** (content printed), **speech** (content for speech synthesis)
- Media features: **height** (of the viewport in px), **orientation** (of the device - landscape or portrait), **width** (of the viewport), etc.
- More info: [https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using media queries](https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries)



University of
St Andrews

Media Queries

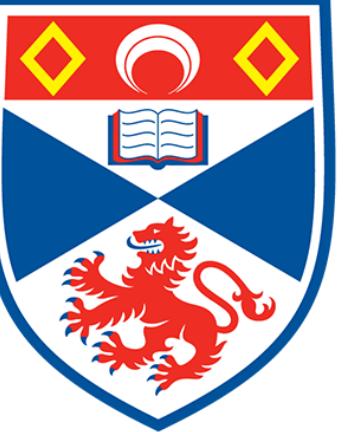
- `@media screen, print { ... }`
- This media query (that targets the media **type**) will apply to both **screen** and **print** devices
- `@media (min-width: 30em) { ... }`
- The above media query (that targets the media **feature**) will apply when the minimum width of a screen is **30ems**.



University of
St Andrews

Advanced Media Queries

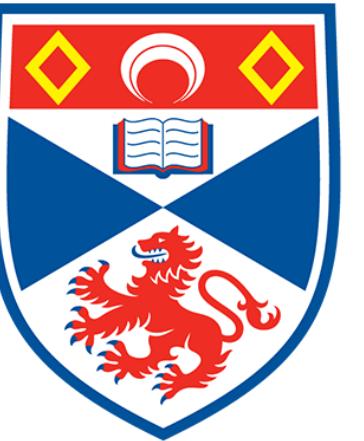
- We can also create complex media queries with multiple statements:
- @media screen and (min-width: 30em) and (orientation: landscape) { ... }
- This media query targets screen devices that have a minimum width of 30ems and are currently in a landscape orientation
- We can also add logic for a **NOT** condition:
- @media (not(hover)) { ... }
- This media query will apply logic to devices that do not have a hover capability (such as a tablet or phone)



University of
St Andrews

Advanced Media Queries

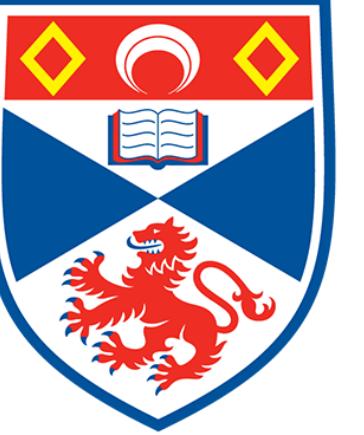
- We can also use **OR** in our media queries as well:
- @media (not (hover)) or (orientation: landscape) { ... }
- This media query will apply to devices that do not have a hover capability or are in landscape mode
- We can also use the **only** keyword to target only a certain kind of screen, layout, etc.
- @media only screen and (max-width: 600px) { ... }
- We can apply complex and advanced logic with media queries
- All of this allows us to create responsive websites that adapt based on the screen, colour of the screen, how it looks when printed, etc.



University of
St Andrews

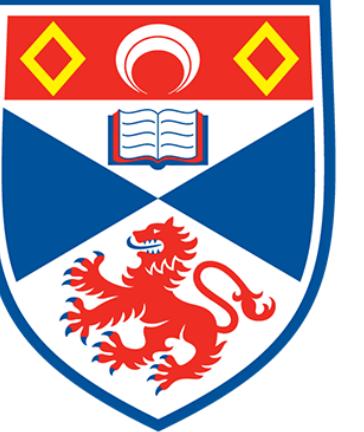
Conditional CSS imports

- We can even make use of **import** in order to import certain CSS based on a condition
- For instance, let's say you wanted to use a different version of your CSS for dark mode on a website
- We can use the **@import** CSS rule
- In our CSS, we would write:
- `@import "cdn.com/my-css-light.min.css" screen;`
- `@import "cdn.com/my-css-dark.min.css" screen and (prefers-colour-scheme: dark);`



University of
St Andrews

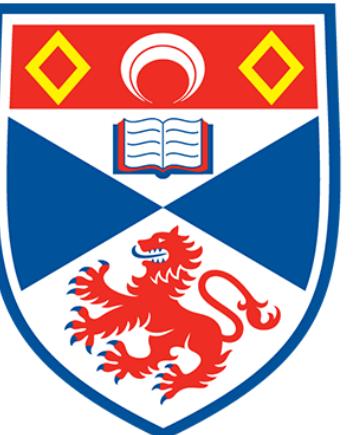
Grid Layout



University of
St Andrews

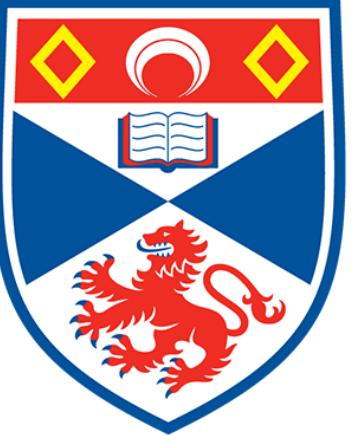
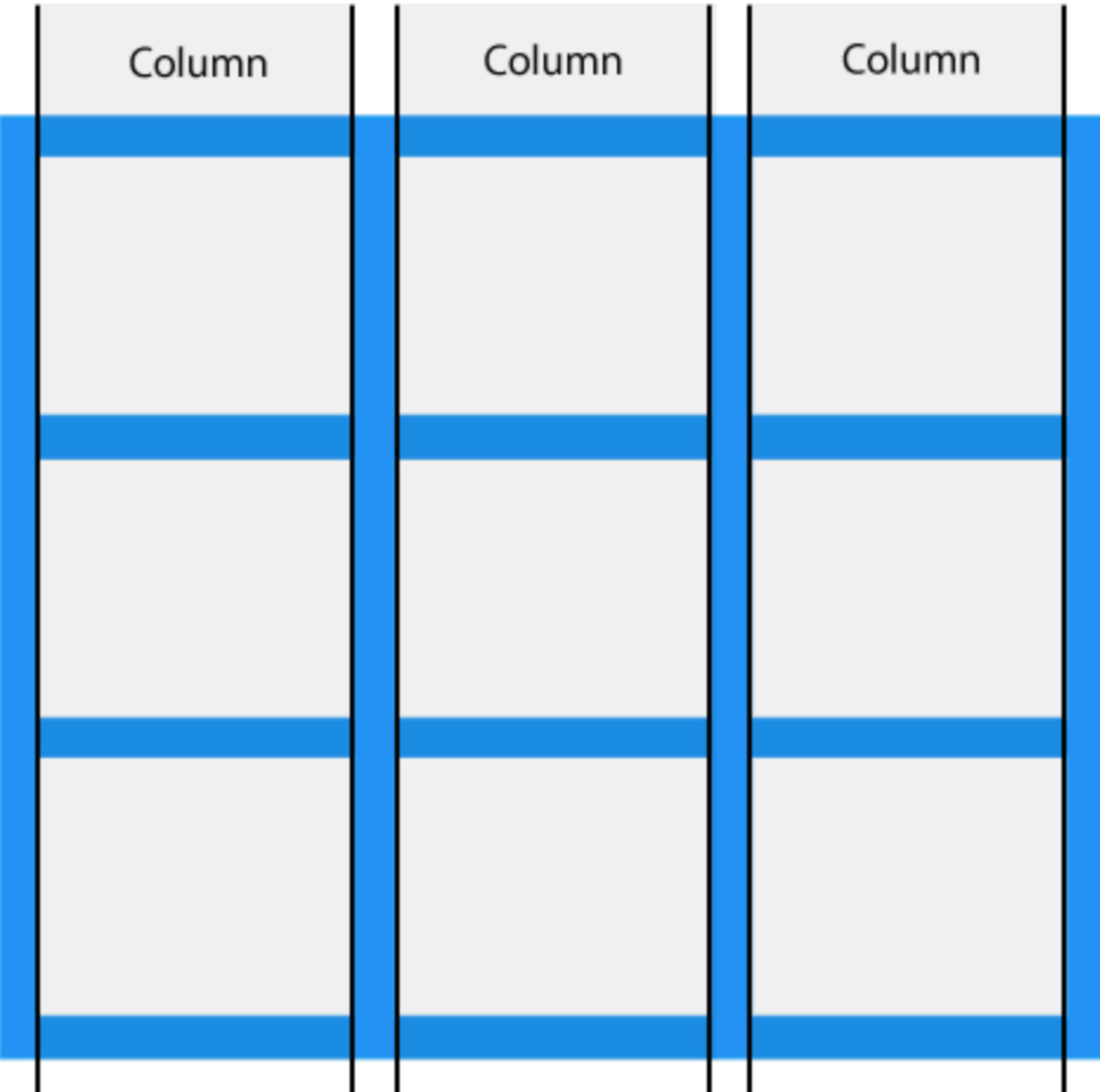
Grid Layout

- We can also arrange our content into a specific layout that can allow for even more flexibility when dealing with various devices
- A page is split into **rows** and **columns**

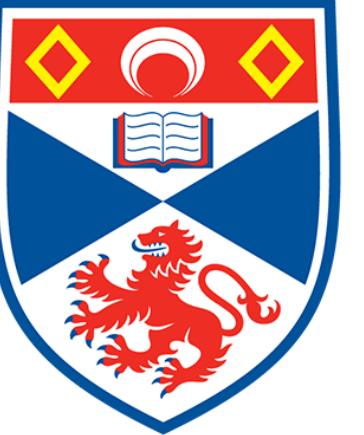


University of
St Andrews

Row				
Row				
Row				



University of
St Andrews

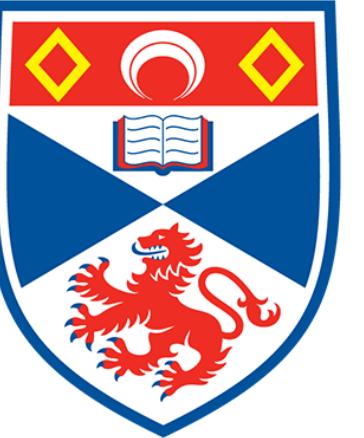


University of
St Andrews

Grid Layout

- We specify an element that acts as a **grid container**
- All elements contained within the grid container become grid items
- There are several selectors we apply to build our grid
- For example: **grid-template-columns** specifies the number and widths of our various columns
- **grid-template-rows** specifies the number and widths of our various rows, etc.
- All of these allow us to build a flexible grid layout

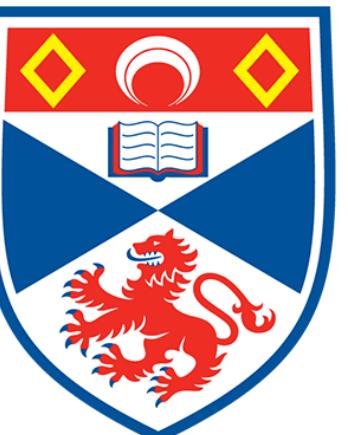
1	2	3
4	5	6
7	8	9



University of
St Andrews

Grid Layout

- In the grid container, we can also specify the gap between columns and rows
- **.grid-container {display: grid; grid-column-gap: 50px; grid-row-gap: 50px;}**
- All of these allow us to build more custom grids
- Source and more on grids: https://www.w3schools.com/css/css_grid.asp



University of
St Andrews

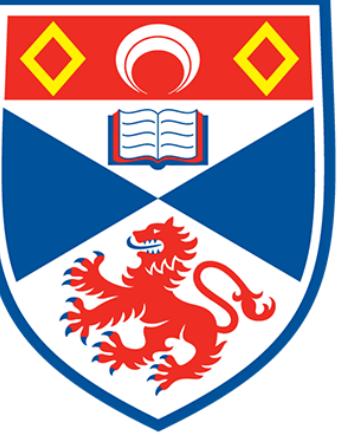
1	2	3
4	5	6
7	8	9



University of
St Andrews

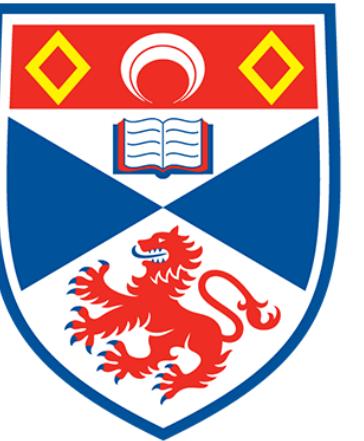
Viewports, Media Queries and Grids

- We can combine the use of grids, media queries and adapting our content based on the device in order to create flexible and responsive websites that work across a range of devices
- Creating complex websites that are adaptive to various devices becomes much easier using these approaches



University of
St Andrews

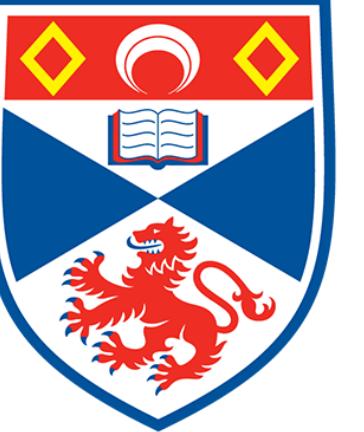
**It didn't always use to be this
way...**



University of
St Andrews

Adaptive Website Design (AWD)

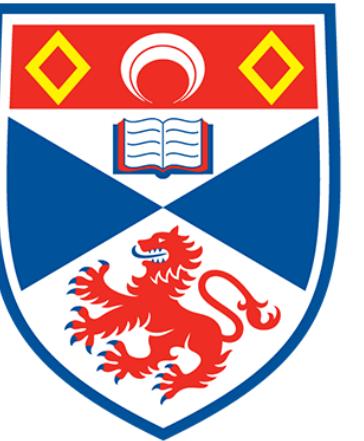
- Before the introduction of grids, media queries, etc. developers would create two versions of a website (one for web and one for mobile)
- You may have noticed some websites accessed on your mobile start with: <http://m.mywebsite.co.uk>
- This signified that you were accessing the mobile version of the website
- This was called **adaptive web design**
- With adaptive web design, one would create different versions of their site based on the screen width of the device (normally around 6 for various breakpoints)
- You would then load the various versions for the device the user was accessing your site from
- These versions **would not respond** - i.e. they had **static layouts**



University of
St Andrews

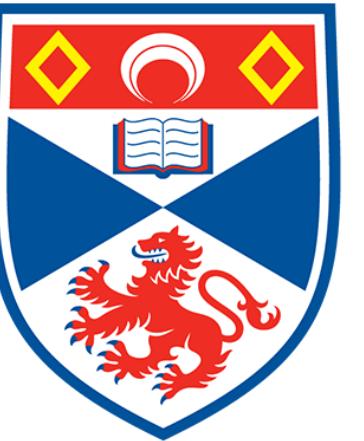
AWD and RWD

- Before websites were created with a web-first mindset - we did not have the range of devices we have now (tablets, phones, TVs, etc.)
- Now modern websites are created mobile-first
- There are advantages and disadvantages to using AWD vs RWD
- This is dependent on many factors - your use case, your codebase, etc.



University of
St Andrews

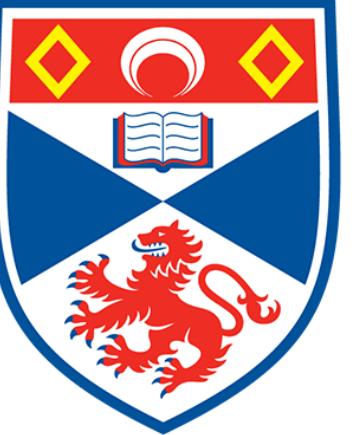
Responsive Media



University of
St Andrews

Responsive Images

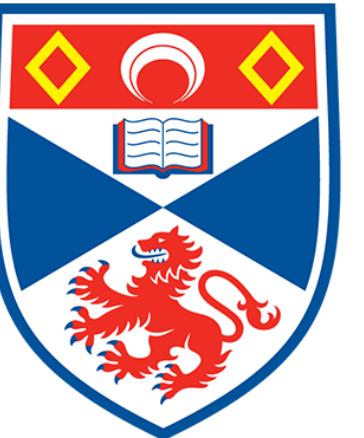
- It's not just text content that needs to be responsive, media content must also be responsive
- Again, various end-user devices have different resolutions, screen sizes, etc.
- We want to ensure that our media assets are suitable for each group of devices accessing our site
- Remember accessibility!



University of
St Andrews

Responsive Images

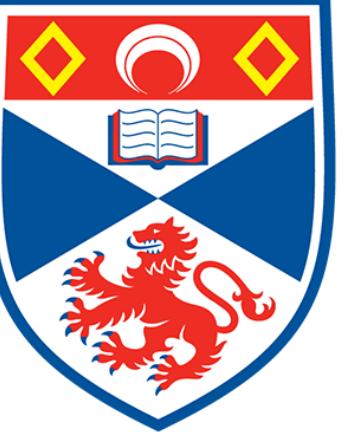
- We can use the **srcset** and the **sizes** attributes which will allow the browser to pick the right resolution of image based on the end user device
- **srcset** lists the set of images that the browser can choose from (same image, various sizes and resolutions)
- **sizes** defines a set of conditions (for the device) for which it should select specific image(s) listed in the **srcset**



University of
St Andrews

Responsive Images

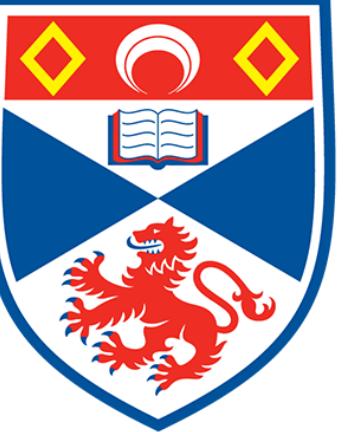
- ``
- This effectively says pick the my-image-480w.jpg file if the current screen size has a max width of 600px (i.e $\leq 600\text{px}$), else if not then remain with the 800w image.



University of
St Andrews

Responsive Images

- We can also use simpler CSS to “make” an image responsive
- We can add CSS to scale our image up and down
- `.responsive { width: 100%; height: auto; }`
- Our image will scale appropriately based on the window



University of
St Andrews

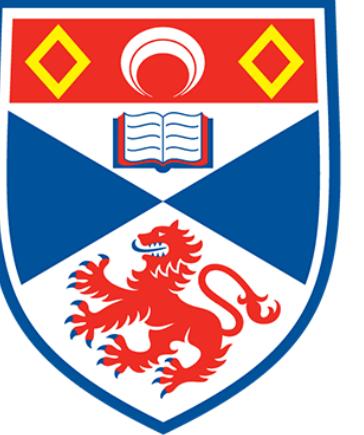
Responsive Video

- We can do the same to a video
- We can add CSS to scale our image up and down
- `.responsive-video { width: 100%; height: auto; }`
- Our video will scale appropriately based on the window
- We can also do the same thing if using a `<video>` element
- `<video width="100%" ... ></video>`
- The aspect ratio of the video is maintained as it responds (important not to set a height attribute for this to occur)



University of
St Andrews

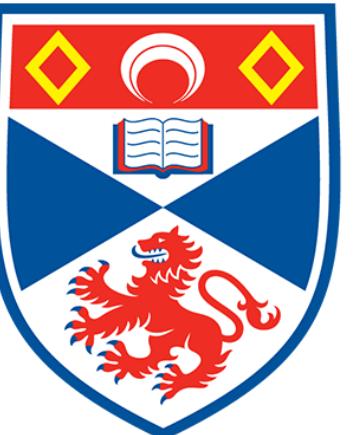
Frameworks



University of
St Andrews

Frameworks

- There are several frameworks that can help you to quickly create responsive websites
- Bootstrap is a very popular framework, we also have other frameworks such as Foundation, etc.
- It comes with responsiveness built in by default, also includes aspects like grid layouts, responsive media, etc.
- Bootstrap's grid layout is also based on columns and rows
- There are **12** columns in a bootstrap row
- You can simply declare something like `<div class="col-3"></div>` and the div's content will fill 3 columns.



University of
St Andrews

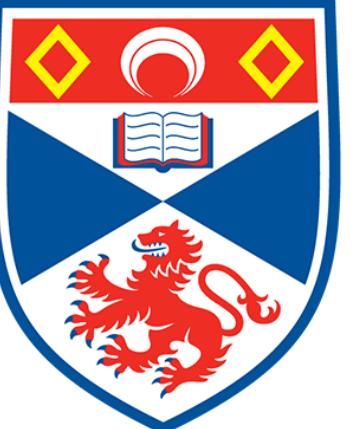
Frameworks

.g-col-4 .g-col-4 .g-col-4

```
<div class="grid">
  <div class="g-col-4">.g-col-4</div>
  <div class="g-col-4">.g-col-4</div>
  <div class="g-col-4">.g-col-4</div>
</div>
```

Copy

Source: <https://getbootstrap.com/docs/5.1/layout/css-grid/>



University of
St Andrews

Frameworks

.g-col-6 .g-col-md-4

.g-col-6 .g-col-md-4

.g-col-6 .g-col-md-4

```
<div class="grid">
  <div class="g-col-6 g-col-md-4">.g-col-6 .g-col-md-4</div>
  <div class="g-col-6 g-col-md-4">.g-col-6 .g-col-md-4</div>
  <div class="g-col-6 g-col-md-4">.g-col-6 .g-col-md-4</div>
</div>
```

Copy

Compare that to this two column layout at all viewports.

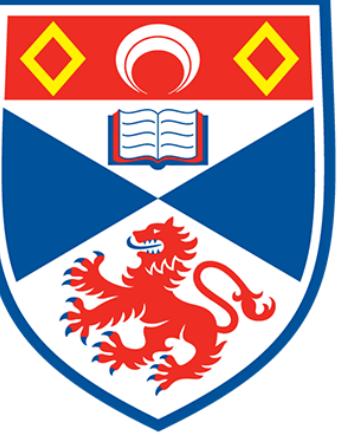
.g-col-6

.g-col-6

```
<div class="grid">
  <div class="g-col-6">.g-col-6</div>
  <div class="g-col-6">.g-col-6</div>
</div>
```

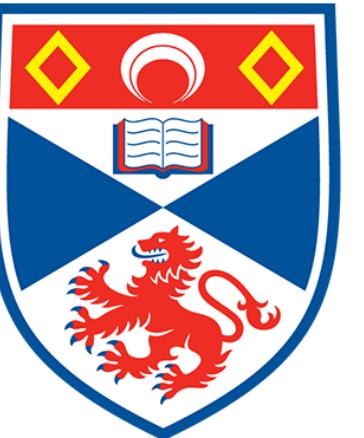
Copy

Source: <https://getbootstrap.com/docs/5.1/layout/css-grid/>



University of
St Andrews

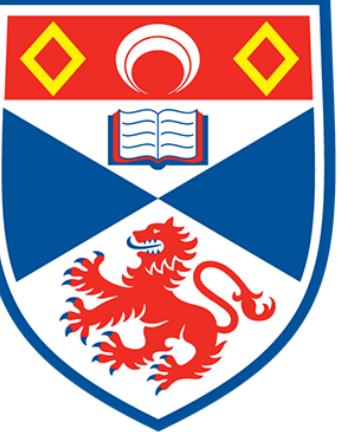
Testing



University of
St Andrews

Testing responsive website

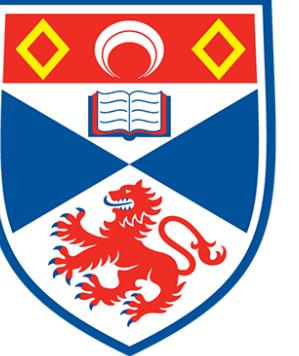
- We need to test our sites to ensure they work on various device screen widths and they respond accordingly when they reach the various breakpoints
- We can do this on our own devices or also on emulators
- Mobile friendly test: <https://search.google.com/test/mobile-friendly>
- More tools: <https://www.creativebloq.com/features/7-great-tools-for-testing-your-responsive-web-designs>



University of
St Andrews

More reading

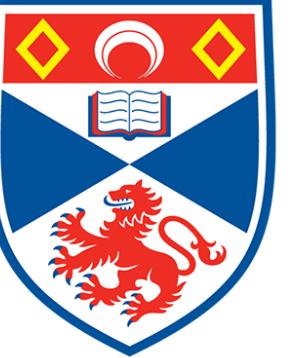
- Responsive images: [https://developer.mozilla.org/en-US/docs/Learn/HTML/Multimedia_and_embedding/Responsive images](https://developer.mozilla.org/en-US/docs/Learn/HTML/Multimedia_and_embedding/Responsive_images)
- Responsive Video: <https://css-tricks.com/fluid-width-video/>



University of
St Andrews

Forms and Interactivity

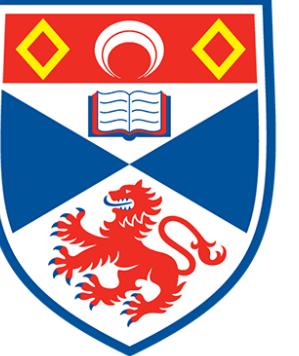
Nnamdi Ekwe-Ekwe



University of
St Andrews

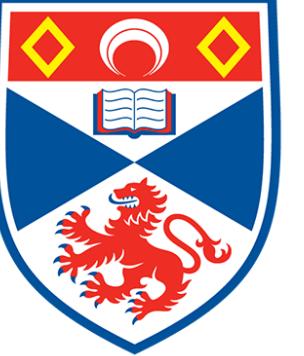
Taking stock

- Where are we now?
- Web Standards
- Markup - HTML, XHTML, HTML5
- HTML + CSS
- “HTML describes, CSS styles”
- Accessibility
- Responsive Web



University of
St Andrews

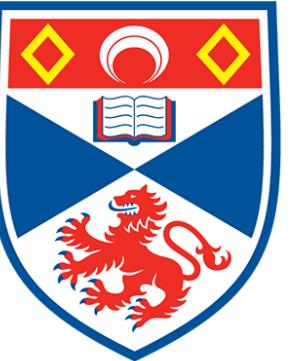
Where next?



University of
St Andrews

Interactivity

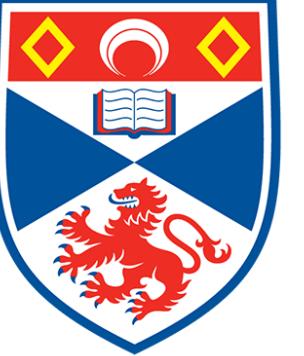
- Interactivity is an essential part of any website
- Gone are the days of “static” websites that just presented information
- Users interact with websites in a variety of different ways
- Authentication, personalising their experience, changing settings in their account, all of these different aspects require an interactive website
- A critical element of an interactive experience on a website is the **form**



University of
St Andrews

The image shows two side-by-side screenshots of login interfaces. On the left is the 'Sign in to iCloud' screen, which features a light blue and yellow gradient background. It has a white cloud icon at the top, followed by the text 'Sign in to iCloud'. Below this is a text input field with 'Apple ID' placeholder text and a right-pointing arrow button. Underneath the field is a checkbox labeled 'Keep me signed in'. At the bottom is a link 'Forgotten your Apple ID or password?'. On the right is the 'Google' search homepage, which has a dark grey background. The word 'Google' is prominently displayed in white at the top center. Below it is a search bar with a magnifying glass icon. At the bottom of the search bar are two buttons: 'Google Search' and 'I'm Feeling Lucky'.

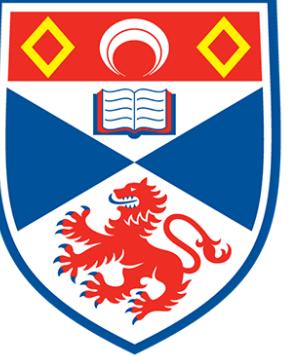
A screenshot of the Amazon.com website's navigation bar. From left to right, it includes: a dropdown menu with 'All' selected; a search bar with an orange search icon; a dropdown menu with the United Kingdom flag; a greeting 'Hello, Sign in'; a 'Account & Lists' dropdown; a 'Returns & Orders' link; and a 'Basket' icon with a small orange '0' indicating no items.



University of
St Andrews

The input

- The three text boxes we saw on the previous slide are the **input** to the form
- The **input** allows you to type some data
- **<input type="text"/>**
- The above tag will create an empty text field allowing you to type text
- Note that there is **no closing tag** with an **input** - it's **self-closing**
- **<input></input> (Not correct)**
- **<input/> (Correct)**



University of
St Andrews

Input types

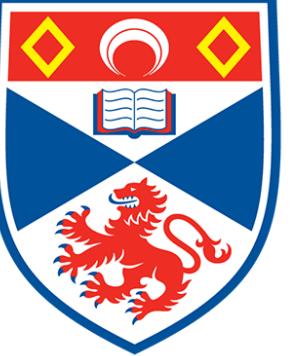
- There are several input **types**.
- These types modify the input tag in order to meet different scenarios:
- Password:
 - `<input type="password" id="pass" name="password"/>`
- Email:
 - `<input type="email" id="email" name="email"/>`
- When one types in a password into the password input, the text will be hidden (displaying as black dots) and for the email, it'll need to contain a "@".



University of
St Andrews

Input types

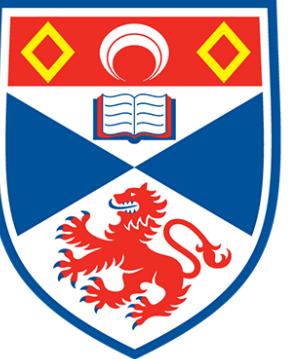
- There are various input types from email and password to date, file, colour, etc.
- All of these different input types can be used as needed for specific scenarios.
- For example, a website developer may want a user to specify their date of birth - a **date** input type would then be specified for the form
- Alternatively a website may require a user to upload their picture - a type of **file** would then be specified in order to accept this file from the user



University of
St Andrews

Input types

- As HTML has advanced, the various input types have advanced too
- Various browsers may render some of these input types differently
- Some browsers won't render them at all
- For example, Internet Explorer 11 does not recognise an `<input type="color">`
- Feeding back to our work on accessibility, it is important to ensure that whatever input type you use is supported across the users' browser set you're catering for



University of
St Andrews

Other input attributes

- There are several other input attributes apart from **type** that we use
- We could specify a **placeholder** attribute that will add a placeholder to a text field before text is input

```
<input type="email" name="email" placeholder="Please enter your email..." />
```

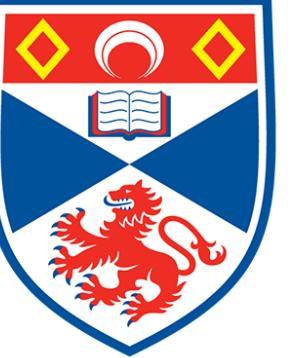
- If a field is **required** then we can use the **required** attribute

```
<input type="email" name="email" required/>
```

- Returning to the password type again, we can specify a **minlength** that will ensure that a password field will have a minimum length of n characters

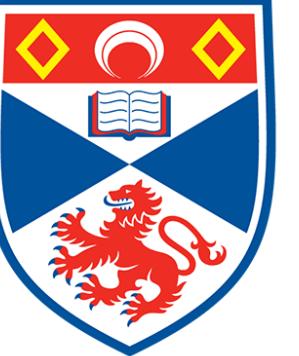
```
<input type="password" name="password" minlength="8"/>
```

- There are several other attributes that can be used for the input, these are just some of them



University of
St Andrews

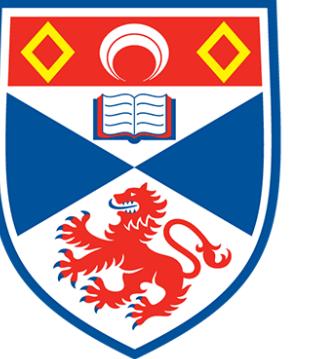
Where do we go from here?



University of
St Andrews

Example

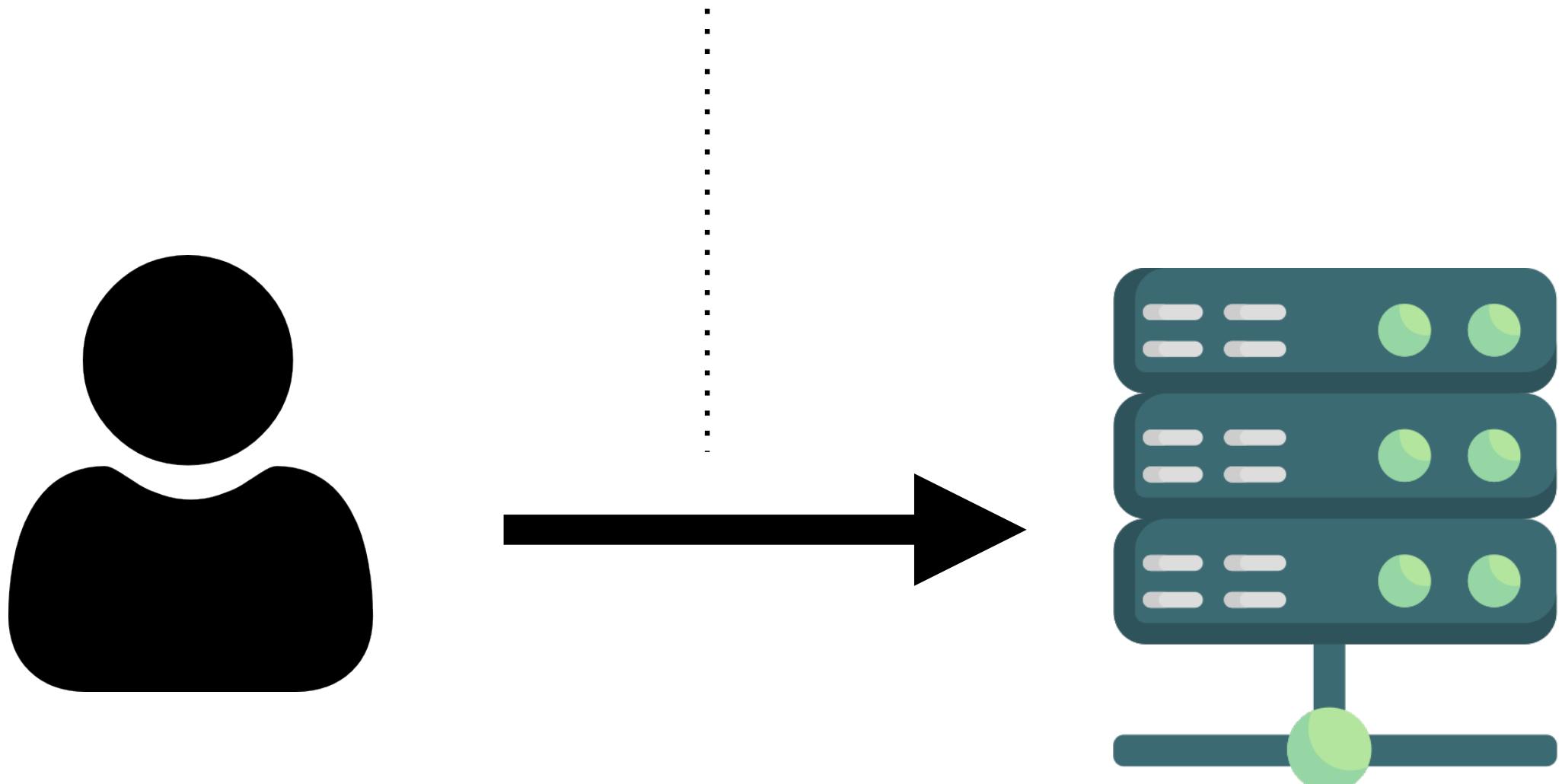
- ...
<input type="email" name="email" placeholder="Please enter your email..." />
<input type="password" name="password" placeholder="Please enter your password..." minlength="4" maxlength="90" />
...

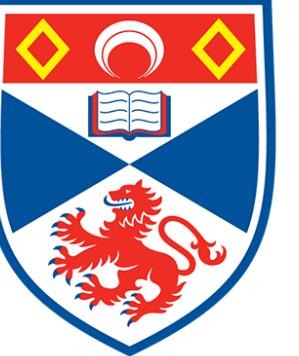


University of
St Andrews

email=“joebloggs@joebloggs.co.uk”

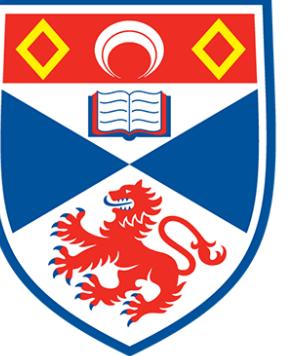
password=“seriouslysecurepassword”





University of
St Andrews

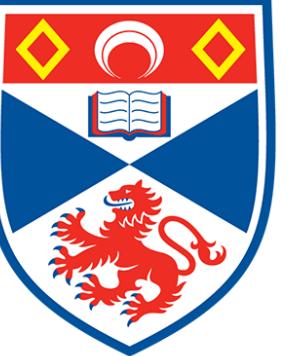
The Form



University of
St Andrews

The form

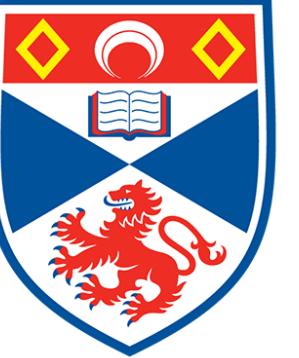
- An input by itself cannot do anything
- It needs to be combined with a **<form>** in order for it to have an effect
- The interactive **input** controls within a **form** is what enables data to be sent to achieve some desired effect
- Forms, as with inputs have their own attributes that need to be specified



University of
St Andrews

Methods

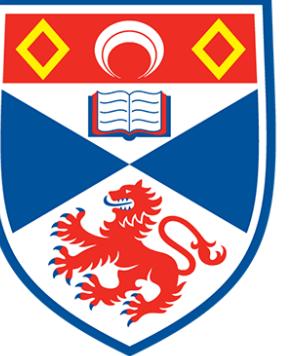
- There are four HTTP Methods (**POST**, **PUT**, **GET**, **DELETE**) which characterises each HTTP operation
- An HTTP operation will either create, read, update or delete some data (**CRUD**)
- For forms, we are interested in only two methods - **POST** and **GET**
- **POST** sends data in the request body to a server
- **GET** retrieves some data from the server



University of
St Andrews

GET VS POST

- You will know a GET request is taking place because of an ? in the URL
- For example: <https://www.google.com/search?q=w3c&client=desktop>
- This is a GET request - we have a ? - in this case the query is “w3c” on Google
- With a POST request however, no added information is in the URL - this is sent “transparently” as part of the request body
- Whenever you’re sending confidential information, don’t use GET!
- Imagine:
 - [https://www.google.com/login?
username=joebloggs&password=seriouslysecurepassword](https://www.google.com/login?username=joebloggs&password=seriouslysecurepassword)



University of
St Andrews

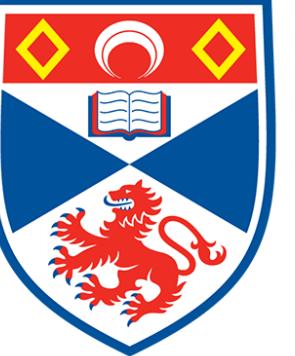
Forms

- When we write a form, we must specify the method
- <form method="POST">

...

</form>

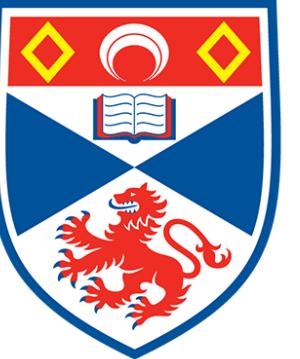
- The **action** attribute specifies the URL that you want to process the form data.
- There would normally be some server running that will accept the form data and process it



University of
St Andrews

Example form attributes

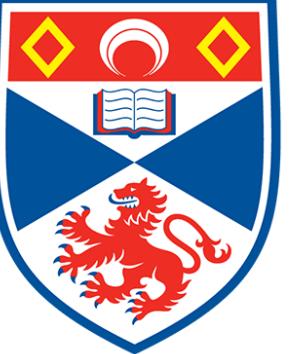
```
<form method="POST" action="/api">
  <input type="email" name="email" placeholder="Please enter your email..." />
  <input type="password" name="password" placeholder="Please enter your
    email..." />
  <input type="submit" />
</form>
```



University of
St Andrews

Improving our form

- We can improve upon our form by labelling the various inputs
- For example:
- `<label> <input type="email" name="email"/> Email: </label>`
- `<label> <input type="password" name="password"/> Password: </label>`
- Remember good markup!



University of
St Andrews

Improving our form

- We can additionally make use of elements such as legend and fieldset to make our form more structured
- A **fieldset** groups various input controls whilst a **legend** will effectively label those fieldsets:

```
<!-- Form which will send a GET request to the current URL -->
```

```
<form>
```

```
    <label>Name:
```

```
        <input name="submitted-name" autocomplete="name">
```

```
    </label>
```

```
    <button>Save</button>
```

```
</form>
```

```
<!-- Form which will send a POST request to the current URL -->
```

```
<form method="post">
```

```
    <label>Name:
```

```
        <input name="submitted-name" autocomplete="name">
```

```
    </label>
```

```
    <button>Save</button>
```

```
</form>
```

```
<!-- Form with fieldset, legend, and label -->
```

```
<form method="post">
```

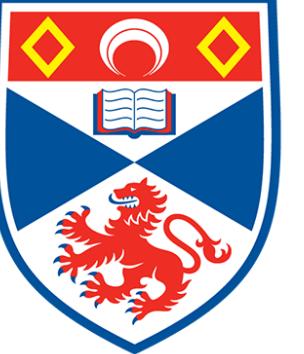
```
    <fieldset>
```

```
        <legend>Title</legend>
```

```
        <label><input type="radio" name="radio"> Select me</label>
```

```
    </fieldset>
```

```
</form>
```



University of
St Andrews

Legend

Name: Save

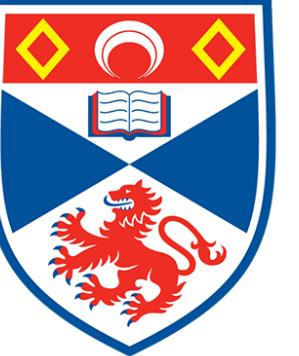
Name: Save

Title

Select me

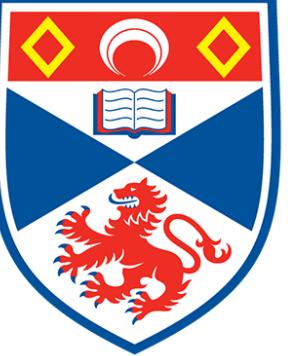


Fieldset



University of
St Andrews

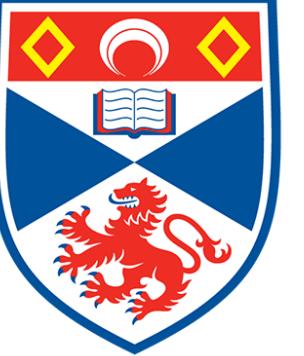
Accesskeys and Tabindexes



University of
St Andrews

Accesskeys

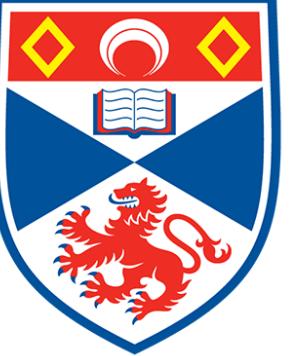
- Keyboard shortcuts:
 - Accesskeys
 - Allow you to access a specific element via a shortcut
 - Can also be used within forms
 - https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes/accesskey
 - You specify the accesskey character within the element and then you use a shortcut to focus on that element
 - <button accesskey='s'>Click me</button>



University of
St Andrews

Disadvantages of accesskeys

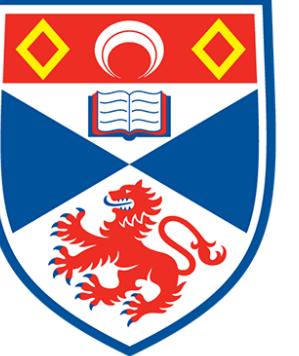
- Accesskeys can interfere with native system keyboard shortcuts
- Poor browser support
- Can also interfere with assistive technologies present on a device
- When you introduce international keyboards, problems can be amplified as various keyboards have various symbols
- Therefore, you have to internationalise this - which can also be more problematic
- Best to avoid using accesskeys
- Remember accessibility!



University of
St Andrews

Tabindexes

- **Tabindexes:**
 - Specifies the order of focus on elements - if you use the tab key you can navigate the various elements on the website in a specific order
 - This can be applied to various elements on a webpage
 - Very useful in forms
 - `<input type="text" name="firstName" size="20" tabindex="1" />`
 - `<input type="text" name="lastName" size="20" tabindex="2" />`
- If you were to use the tab button on your keyboard, you can navigate these input elements in the **specified order** given in the tabindex



University of
St Andrews

Forms 2.0



University of
St Andrews

Forms 2.0

- Forms 2.0 extended the form tag in HTML 4.0
- Forms 2.0 is implemented in HTML5
- Offers different new values for the **type** attribute in a form input tag such as date, range, url, etc.
- Also introduced new attributes such as **placeholder**, **output**, **required**
- More reading: https://www.tutorialspoint.com/html5/html5_web_forms2.htm

Form validation

- HTML5 forms come with some built in form validation as standard
- We don't have to rely on JavaScript
- We can validate on both the client-side as well as the server-side - this is best practice
- When an element is not valid, for example an email does not have an “@” symbol or a password doesn't include a specified pattern, we can display this error to the user using CSS pseudo classes (discussed later)

Form HTML

```
<fieldset>
  <h2 class="account">Account</h2>
  <ul>
    <li>
      <label for="first_name">First Name:</label>
      <input type="text" id="first_name" name="first_name" class="large"
             placeholder="Enter your first name" required />
    </li>
  </ul>
</fieldset>
```

Form CSS

```
input:valid {  
    border-color: green;  
    border-style: solid;  
    border-width: 2px;  
}  
  
input:invalid {  
    color: red;  
}  
  
input:invalid:required {  
    border-color: red;  
    border-style: dashed;  
    border-width: 2px;  
}
```

The result

VALID

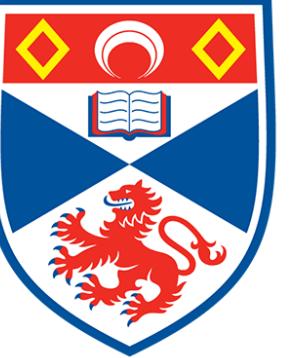
Account

- First Name:

INVALID

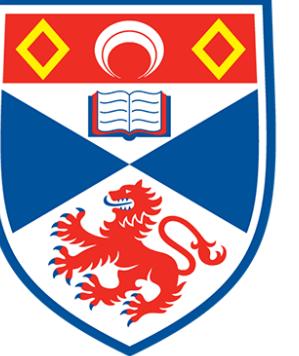
Account

- First Name:



University of
St Andrews

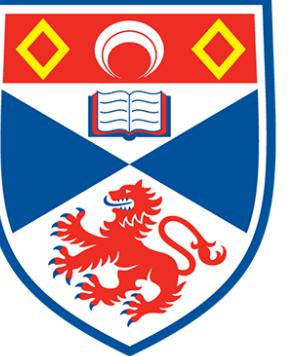
CSS for forms



University of
St Andrews

CSS for forms

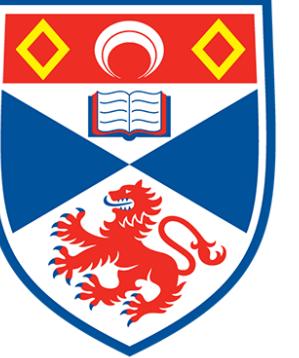
- When styling our forms, we can use very targeted and precise CSS
- We can also make use of CSS **pseudoclasses** allowing us to target specific **states** of the element
- For example, let's say we wanted to change a button in our form to green when we hover on it:
- `<button class="myButton">Hello!</button>`
- `button:hover{
 color:green;
}`



University of
St Andrews

CSS for forms

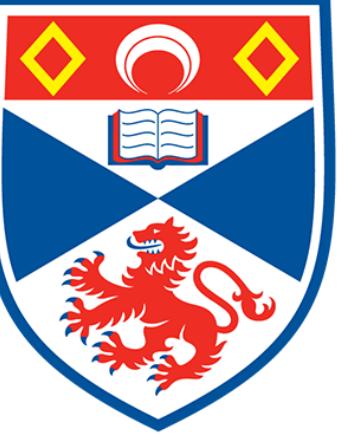
- We can also target specific types of input in our forms too:
- ```
input[type=text] {
 width: 100px;
 background-colour: green;
 color: white;
}
```
- This will give you an input field with a green background and white text that has a width of 100px;
- You can also use icons to style your forms



University of  
St Andrews

# More reading

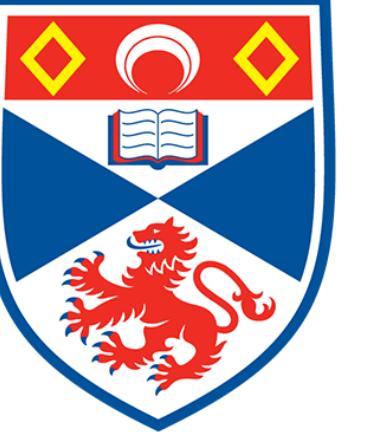
- Styling forms: [https://developer.mozilla.org/en-US/docs/Learn/Forms/  
Advanced form styling](https://developer.mozilla.org/en-US/docs/Learn/Forms/Advanced_form_styling)
- Web Forms 2.0: [https://www.tutorialspoint.com/html5/html5\\_web\\_forms2.htm](https://www.tutorialspoint.com/html5/html5_web_forms2.htm)



University of  
St Andrews

# Web Applications and Frameworks

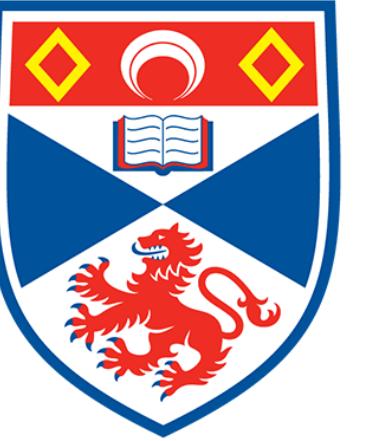
**Nnamdi Ekwe-Ekwe**



University of  
St Andrews

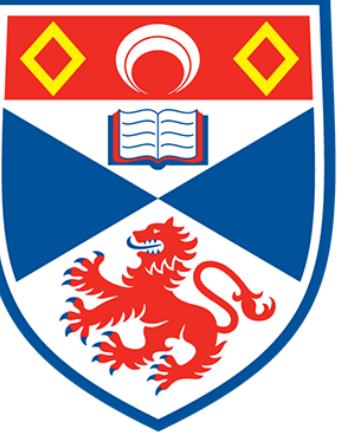
# So far...

- We've looked at:
- HTML
- CSS
- Accessibility
- Forms
- Responsive Web



University of  
St Andrews

# Web applications



University of  
St Andrews

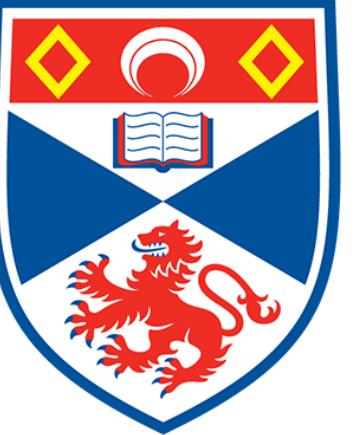
# Some web apps...



**PayPal**™



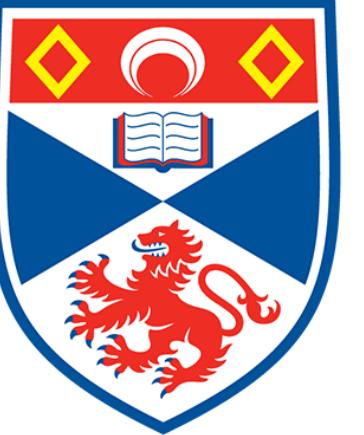
**NETFLIX**



University of  
St Andrews

# Web application vs websites

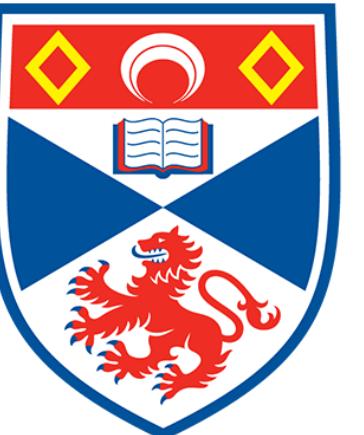
- A website is **static**, does not allow for user interaction/manipulation of data, etc.
- Examples: some sort of information feed, presenting a list of links to you, etc.
- A web application is inherently **interactive**, allowing the user to manipulate data, perform a wide range of functionality (logging in, creating accounts, posting updates to social media, etc.)
- Websites are static, web apps are dynamic and interactive



University of  
St Andrews

# How is this dynamism achieved? (1)

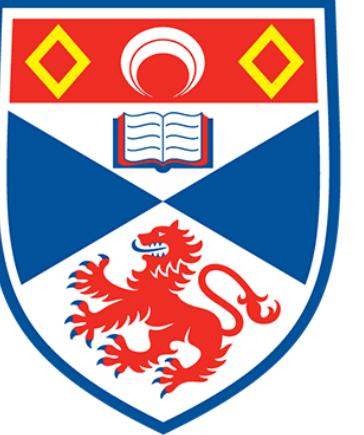
- Your applications must send data to some service, that will perform these functions and update what you see
- Can we do this in HTML/CSS?



University of  
St Andrews

# How is this dynamism achieved? (2)

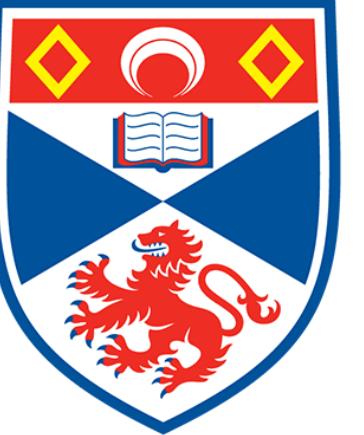
- HTML + CSS are only responsible for the **view** that we see and to allow a means of user interaction with our web application
- However, the underlying functionality has to be implemented by another language or framework
- “HTML presents, CSS styles”



University of  
St Andrews

# The model-view-controller (MVC)

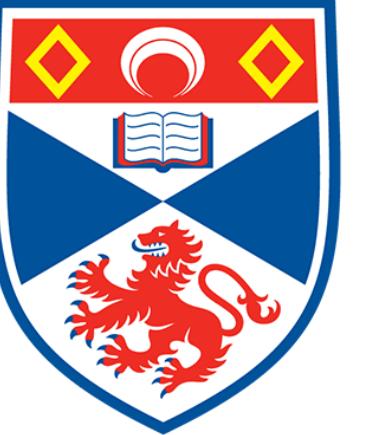
- The MVC concept is at the heart of many web applications
- The web application is split up into **three** logical components/tiers:
  - The model (the data tier)
  - The controller (the business logic tier)
  - The view (the presentational tier)



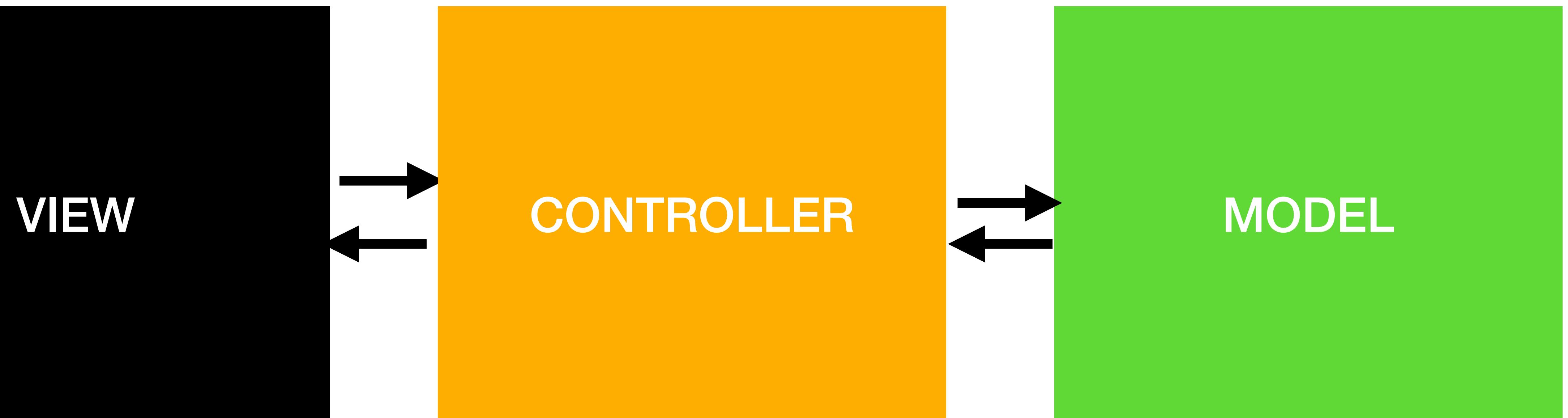
University of  
St Andrews

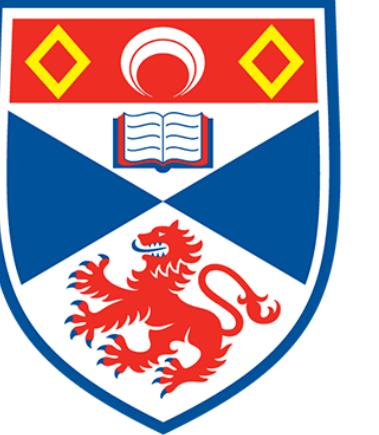
# The model-view-controller (MVC)

- Our data resides in the model/data tier
- The view is our presentation component (the UI) which the user interacts with directly
- These interactions pass to the controller which sits in between the view and the data
- The controller is responsible for carrying out the functionality corresponding to user input and updating the data tier and corresponding view



University of  
St Andrews

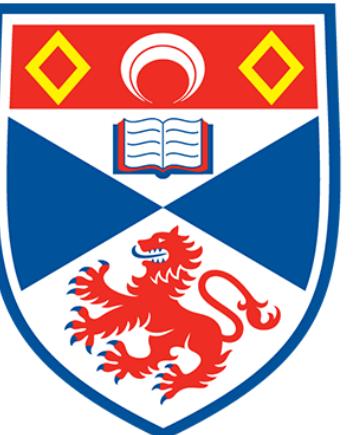




University of  
St Andrews

# An example...

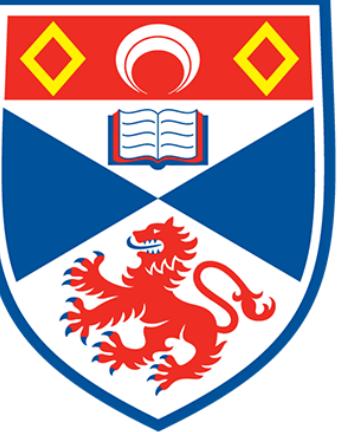
- Take an e-commerce site
- The model (our data) could be our database of products in our store
- The view (our website) allows us to add a product to the cart
- The controller (sitting in between our view and model) receives a user input “add one of Product A to cart”, updates the model appropriately, and then send back the updated product availability back to the view.



University of  
St Andrews

# So how does this relate back to web?

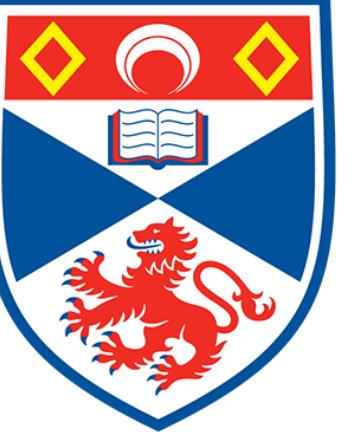
- Thus far, we have only been concentrating on the **view** (HTML, CSS, responsiveness, etc.)
- However, in actual fact, a web application is much bigger than just the view, and is comprised of many different components that leads to our final product.
- In the working world, we would normally have much more than just a knowledge of HTML and CSS, but be able to combine these into a web application framework that allows us to build our web app



University of  
St Andrews

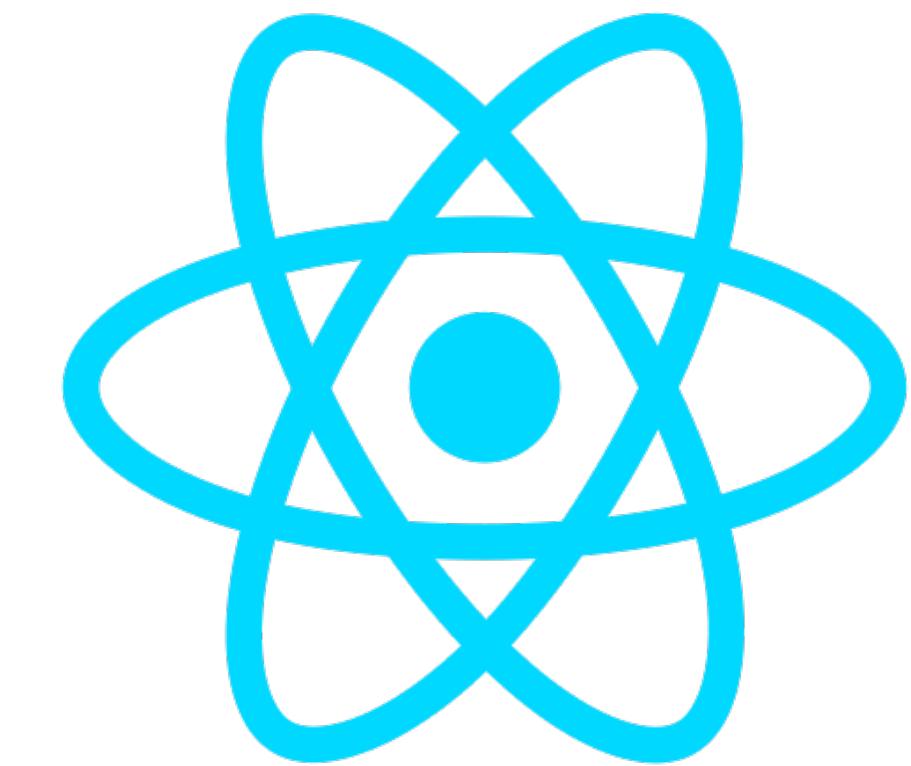
# In real life...

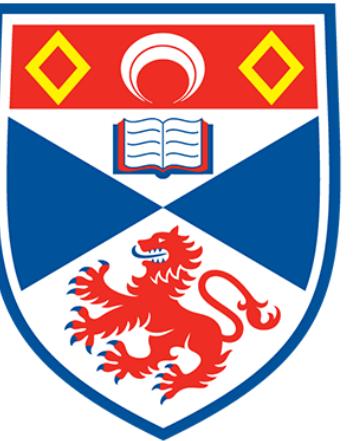
- In the real world, web applications comprise of 1) our what we see on our browser and 2) different background services that accept requests sent from our browser
- These services are what perform the underlying functionality of our web application
- We normally split up our applications into our frontend (presentation) and backend (logic)
- The way we communicate from frontend to backend is via the use of an Application Program Interface or an API



University of  
St Andrews

# Some web frameworks

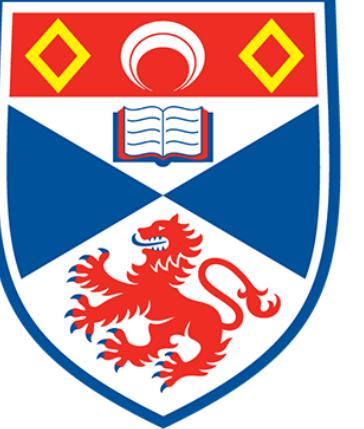




University of  
St Andrews

# Some web frameworks

- These frameworks (Angular, Vue and React) allow you to build fully fledged web applications in an MVC style
- The view (HTML) deals with the presentation
- The controller manipulates data in the model and updates the view
- The model is responsible for storing our data
- These are all built in JavaScript and combine the use of HTML, CSS and knowledge of the framework to build a fully fledged application



University of  
St Andrews

# Angular as an example...

```
1 <h2>Hero List</h2>
2
3 <p>Select a hero from the list to see details.</p>
4
5 <li *ngFor="let hero of heroes">
6 <button type="button" (click)="selectHero(hero)">
7 {{hero.name}}
8 </button>
9
10
11
12 <app-hero-detail *ngIf="selectedHero" [hero]="selectedHero"></app-hero-detail>
13
14
15 <!--
16 Copyright Google LLC. All Rights Reserved.
17 Use of this source code is governed by an MIT-style license that
18 can be found in the LICENSE file at https://angular.io/license
19 -->
```



# Due as an example...

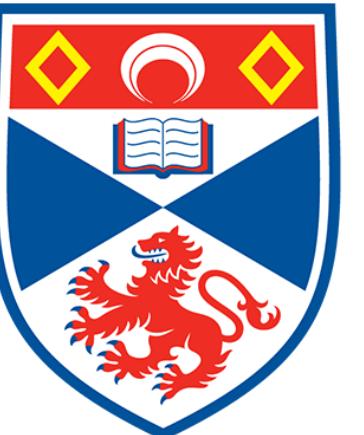
Here is a minimal example:

```
import { createApp } from 'vue' js

createApp({
 data() {
 return {
 count: 0
 }
 }
}).mount('#app')
```

```
<div id="app">
 <button @click="count++">
 Count is: {{ count }}
 </button>
</div>
```

template



University of  
St Andrews

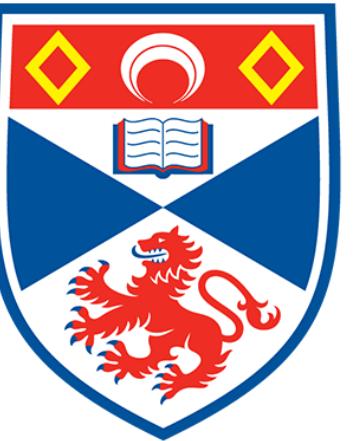
# Some web frameworks

- Frontend developers (concentrating only on the **front-facing** part of the web application) combine their knowledge of HTML, CSS and JavaScript in order to build fully fledged web applications
- These web frameworks help a lot as they allow users to make dynamic and interactive applications with all of the underlying detail abstracted away (hidden) from the developer
- Knowing HTML, CSS and the use of one of these web frameworks will set you in very good stead in the workplace
- Next semester in CS5003, you'll learn about these advanced frameworks/web technologies



University of  
St Andrews

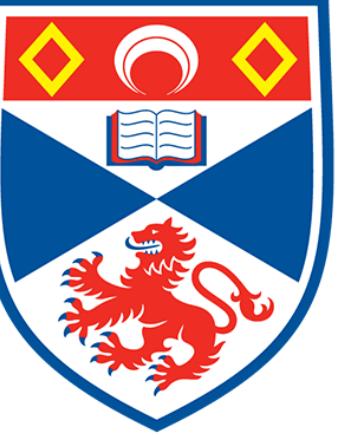
# Application Program Interface (API)



University of  
St Andrews

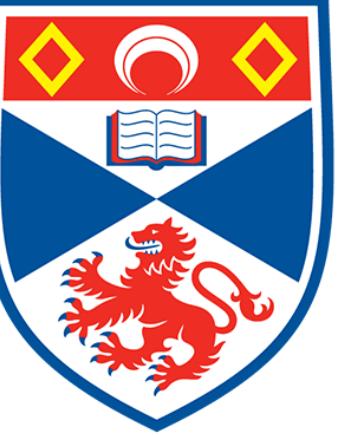
# The Application Program Interface (API)

- Another major aspect that further contributes to our web applications are APIs
- An API is a way of communicating with some service
- You do not know the way the service is implemented in the background, only that it provides some functionality
- You use APIs all the time, both in real life and unknowingly when browsing the web, using apps, etc.
- In this case, our frontend view will communicate with our backend services via some API



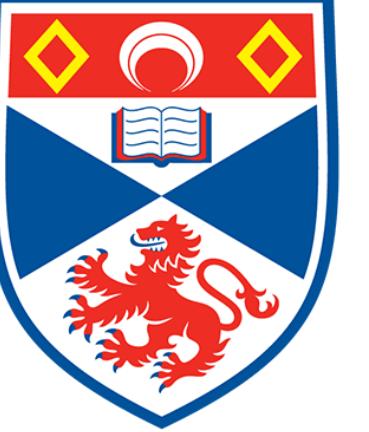
University of  
St Andrews

# Quiz Time



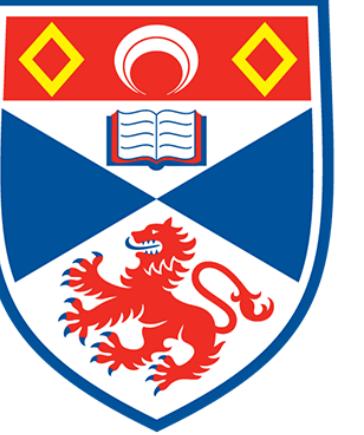
University of  
St Andrews

**What is the best form method for  
these scenarios?**



University of  
St Andrews

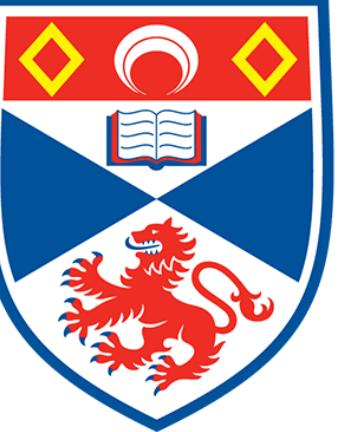
**Q: Creating a user account for a social media website?**



University of  
St Andrews

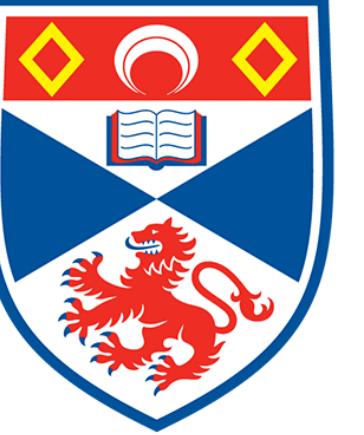
**Q: Filtering out a list of records by two times  
(a start time and an end time)?**

time)



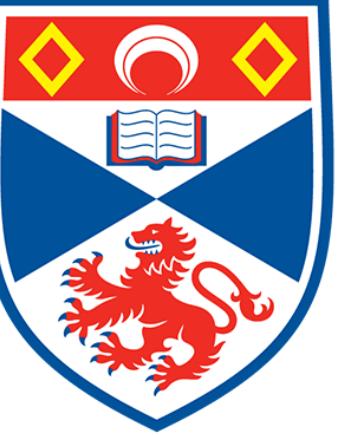
University of  
St Andrews

**Q. Searching for a location on a map?**



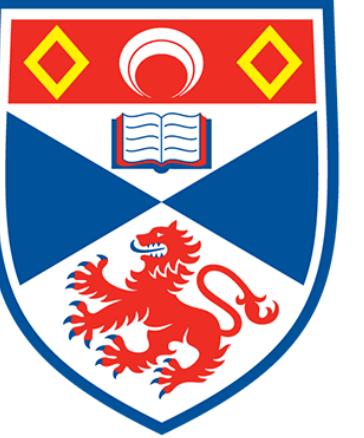
University of  
St Andrews

**Q. Selecting your choice of  
modules for the next semester?**



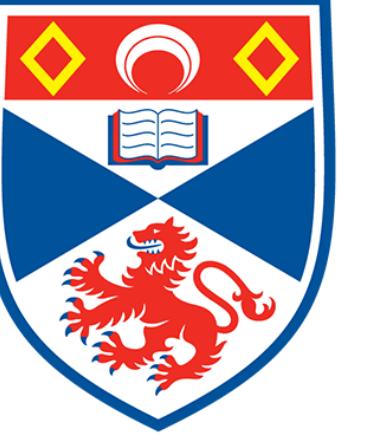
University of  
St Andrews

**Q. Ordering an ice-cream online  
from your local ice-cream shop?**



University of  
St Andrews

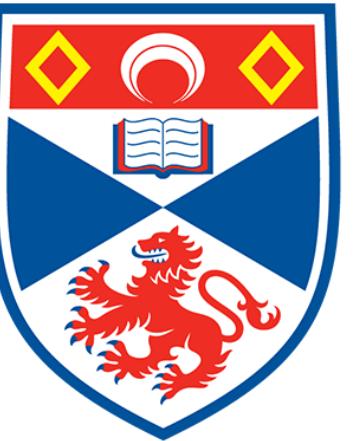
**Write a valid form that will allow a user to search for a name and address of a person. The inputs should be the full name of the person and the address inputs should include the street, town/city, county and country as well as the postcode.**



University of  
St Andrews

# Who wants to be a millionaire?

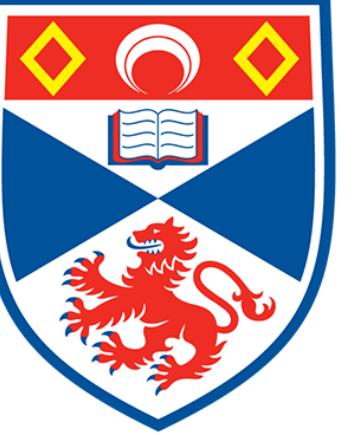
**Nnamdi Ekwe-Ekwe**



University of  
St Andrews

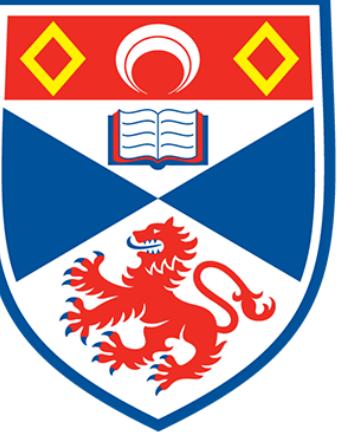
# Group quiz

- We're going to have 5 groups of 9 students each
- We're going to go run the room from group to group answering a range of questions.
- The winning group wins a prize.



University of  
St Andrews

# Round One



University of  
St Andrews

# What does HTML stand for?

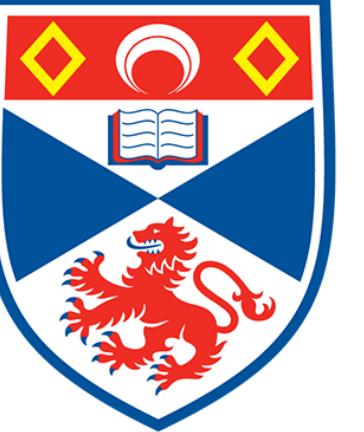
- A. HyperText Markup Language
- B. HyperTexts Markup Languages
- C. HyperReal Markup Linguistics
- D. HyperText Markup List



University of  
St Andrews

# What does CSS stand for?

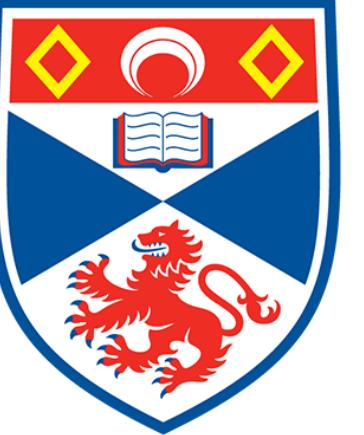
- A. Cascading Stylisation Sheet
- B. Cascading Stylistic Sheet
- C. Cascading Style Sheets
- D. Cascade Style Sheets



University of  
St Andrews

## How would you change the colour of text to red?

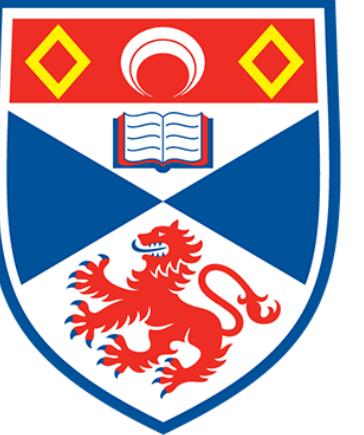
- A. font-color: red;
- B. color: red-color;
- C. color: red;
- D. color-text: red;



University of  
St Andrews

# Which of these tags is invalid?

- A. <h1>Main Header</h1>
- B. <h5>Test Header </h5>
- C. <div><article><video></video></article></div>
- D. <input></input>



University of  
St Andrews

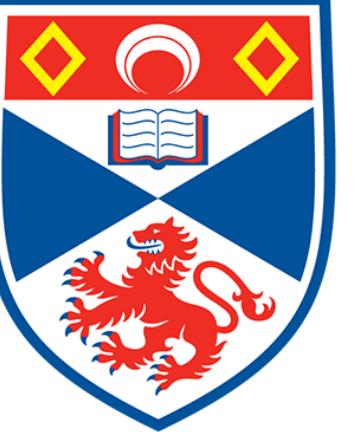
# Consider this code

```

```

What is **wrong** about this code?

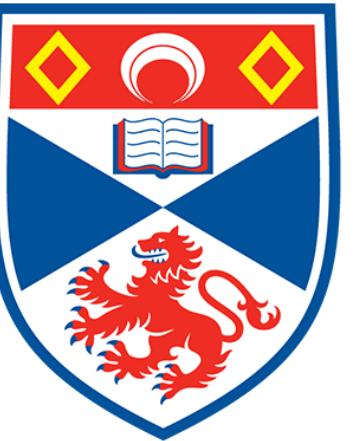
- A. Nothing, this is a correct, well-formed tag
- B. There should be no id for the image as all ids are unique
- C. The src should have single quotes instead of double quotes
- D. The alt tag is not descriptive enough



University of  
St Andrews

In which tag would one place the path to their CSS code?

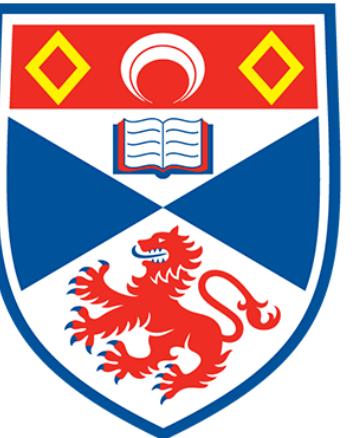
- A. The **rel** tag
- B. The **meta** tag
- C. The **css** tag
- D. The **link** tag



University of  
St Andrews

# How many heading levels are there?

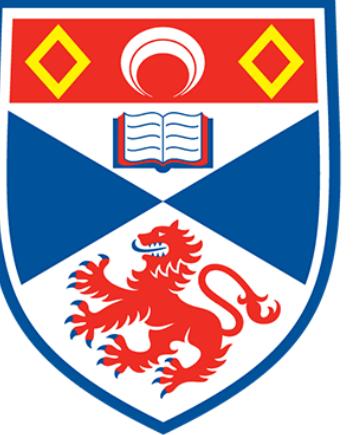
- A. 5
- B. 8
- C. 6
- D. 4



University of  
St Andrews

# How many heading levels are there?

- A. 5
- B. 8
- C. 6
- D. 4

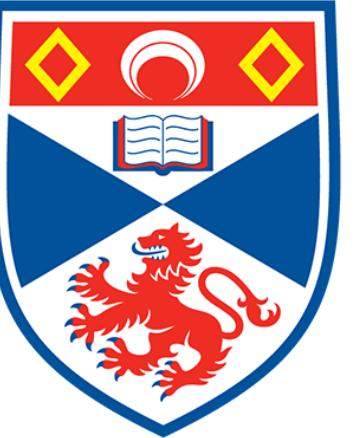


University of  
St Andrews

# Consider this scenario

I've been asked to display a list of all students on my webpage. It doesn't really matter what order they're in, as long as they are visible to the user. What sort of list should I use?

- A. An ordered list as the list of students needs to be visible on the page
- B. An unordered list as the data just needs to be presented without any order
- C. An unordered list as this enables lists to be visible, otherwise they'll be hidden
- D. A table, but with a “list” attribute



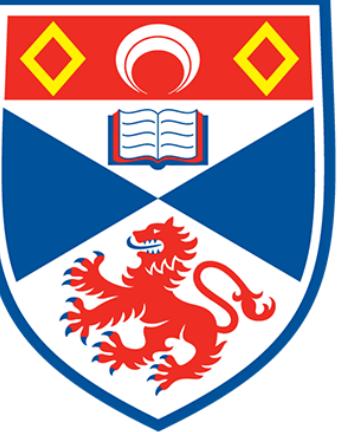
University of  
St Andrews

# Consider this CSS

div .highlight{} and div.highlight{}

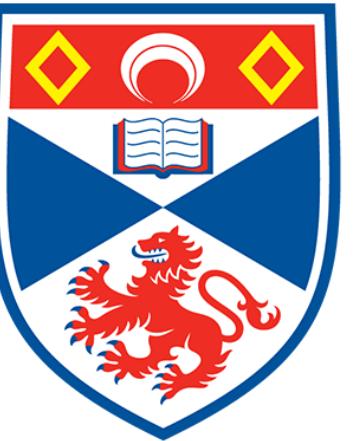
What is the key difference between both these CSS rules?

- A. The first rule will target all divs with a class of highlight, whilst the second will target all elements with a class highlight that are a descendant of the div
- B. The first rule will target all divs or elements that have the class highlight whilst the second rule will target only divs with the class highlight
- C. Both rules are exactly the same, spaces in CSS rules do not make a difference
- D. The first rule will target all elements with the class highlight that are descendants of a div, whilst the second rule will target all divs with a class highlight



University of  
St Andrews

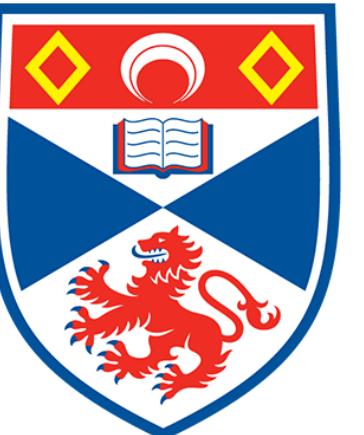
# Round Two



University of  
St Andrews

## What is the difference between AWD and RWD?

- A. AWD is used to create multiple versions of the same website to target different devices, whilst RWD creates two websites to respond to two different scenarios
- B. RWD has one website that responds to multiple screen sizes, whilst AWD has only three websites
- C. RWD is responsive web design, that has one website that responds to multiple screen sizes whilst AWD is where the developer creates multiple versions of a website for different screen sizes
- D. RWD is more focused on mobile devices, whilst AWD is focused on laptops and tablets



University of  
St Andrews

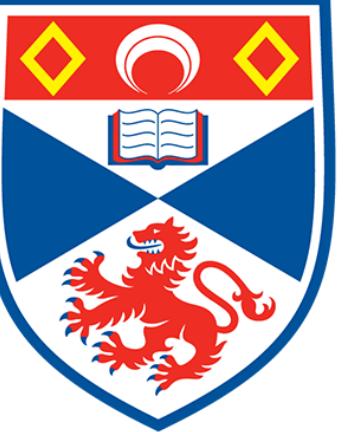
# Consider this CSS

```

```

What would you expect this code to do?

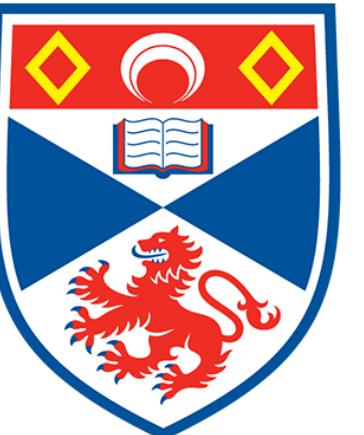
- A. The image that has a width of 900px will be loaded when the screen has a max width of 1000px or else the 1500px image will be used
- B. Nothing, because this is incorrect HTML
- C. The image that's 1500px will be loaded in when we are viewing this website on a screen that is at least 1000px in width
- D. Both images will be used simultaneously until the user selects their correct screen size



University of  
St Andrews

# Which CSS class will make our video responsive?

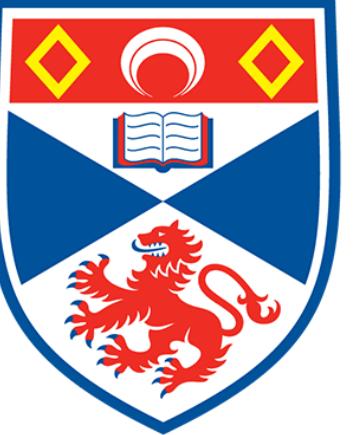
- A. .responsive-video { width: 50%; height: automatic; }
- B. .responsive-video { width: responsive-bootstrap; height: auto; }
- C. .responsive-video { width: 900px; height: auto; }
- D. .responsive-video { width: 100%; height: auto; }



University of  
St Andrews

## Which statement about accesskeys are incorrect?

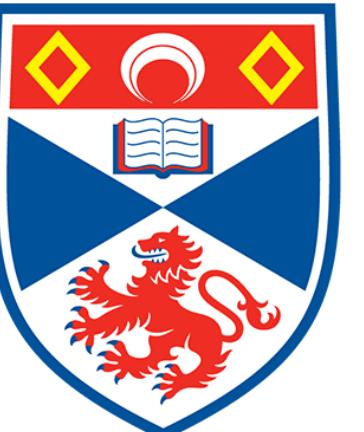
- A. Accesskeys can allow for best practice accessibility on websites
- B. Accesskeys allow for users to specify custom shortcuts to access elements on a website
- C. You specify an accesskey via an **accesskey** attribute in HTML
- D. Accesskeys can be used within forms



University of  
St Andrews

# What is the Robust part of POUR?

- A. “Your website should support and be accessed on a variety devices, including assistive technologies. As technology and user agents evolve, the content must still remain accessible, and be able to be interpreted.”
- B. “Your website should be responsive across all devices”
- C. “Your website should be clear to the user to understand and not pose too high a learning curve for operation.”
- D. “Your website should allow users to login if they have an account”

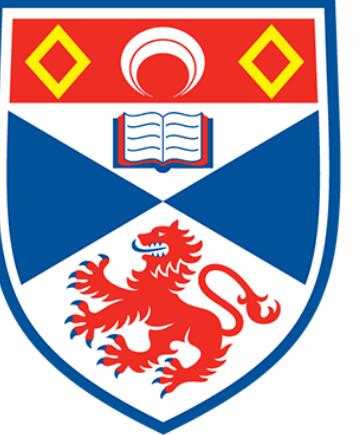


University of  
St Andrews

# Which HTTP request is appropriate here?

Purchasing a new car from a car dealership online

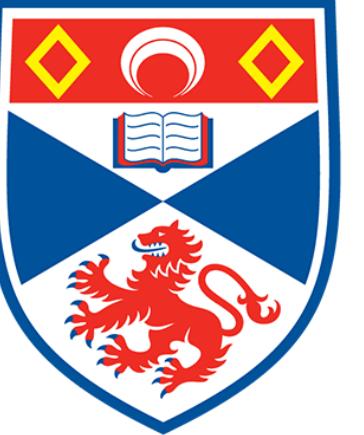
- A. GET
- B. POST
- C. POST and then DELETE
- D. PUT and then POST



University of  
St Andrews

# What is the difference between a web application and a website?

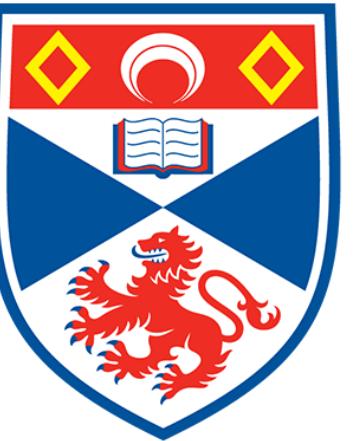
- A. A web application is an MVC framework whereas a website isn't
- B. A web application is what allows for a dynamic and interactive website whilst a website is just focused on HTML + CSS
- C. Web applications are responsive by default, whereas a website is not
- D. Websites allow for MVC whilst web applications are only focused on responsiveness



University of  
St Andrews

## What is the responsibility of the controller in the MVC?

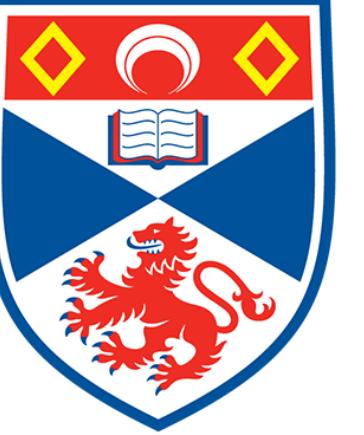
- A. The controller is responsible for just interacting with the model
- B. The controller is responsible for carrying out the functionality corresponding to user input and updating the data tier and corresponding view
- C. User input flows directly to the model which in turn updates the controller
- D. The controller only updates the view, the model is managed by Bootstrap



University of  
St Andrews

# What is an API?

- A. An API provides a gateway between backend services and frontend applications
- B. An API interfaces directly with the MVC to create fast websites
- C. An API is only on the frontend - the backend interacts via the web framework itself
- D. An API is a kind of HTTP Verb



University of  
St Andrews

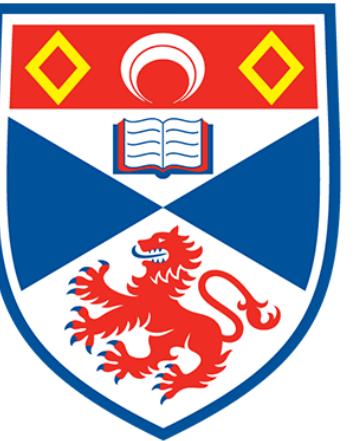
# Which one of these is not exclusively an HTML5 element?

- A. <video>
- B. <aside>
- C. <article>
- D. <div>



University of  
St Andrews

# Round Three



University of  
St Andrews

# Task

- Create a responsive website that contains two pages. The first page should allow a user to signup for a social media website Stbook. Include a question for the user's full name, date of birth, email, password, a password confirmation and an appropriate submit button. Include appropriate validation.
- The second should display an imaginary user's details. These should include the user's full name, date of birth, email and a place to reset their password. This page should span a full 12 columns.
- Use Bootstrap for this website.
- You have 15 minutes.



University of  
St Andrews

# Task (2)

- Create the profile page for your imaginary user.
- This profile page should contain a place for the user's picture and their 5 latest posts (in a sort of timeline-style format). There should also be a navigation bar with three tabs - “Home, Messages, Settings” and a log out button.
- Again, make sure your website is well formed.