```
<!DOCTYPE html>
<html>
    <head>
        <title>digIT 2019 Summer School Workshop</title>
    </head>
    <body>
        <h1 style="text-align: center; font-family: serif;">
```

Building Websites
with
HTML and CSS

```
        </h1>
    </body>
</html>
```

HTML and CSS: An Overview

Hyper**T**ext **M**arkup **L**anguage (HTML) is an XML-based language that provides the **structure** of a web page by "marking up" the content with **tags**. These tags label elements in the page as headings, paragraphs, images, lists and many other types that add structure to content. The HTML standard has undergone several revisions since the first one was released (and is constantly being updated and extended) – the language we will look at in these workshops is HTML 5, released in 2014.

Cascading **S**tyle **S**heets (CSS) is a language used to describe the **presentation** of XML-based documents, most often used for HTML pages. This can include visual properties like layout, colour and fonts, different layouts for different screen sizes and types, and the presentation of pages in different media (for example, text-to-speech). Each version of CSS has been built on the previous version – we will look at CSS 3.

HTML is used for *content*. CSS is used for *presentation*. In a perfect world, our HTML pages convey *document structure and meaning* while our visual design is defined entirely by CSS. Why?

➢ **Visual accessibility** – users can load their own stylesheets
➢ **Flexibility** – responsive design
➢ **Reusability** – external stylesheets
➢ Can you think of other reasons? (Hint: look up "separation of presentation and content"!)
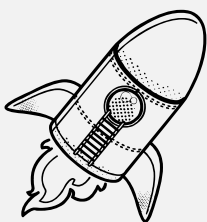
Coding in HTML and CSS probably isn't "computer programming" as you traditionally think of it, but it can be a complex and rewarding pursuit that adds an extra layer to your backend development – as well as incredibly frustrating when you're trying to get every element to sit perfectly in place!

HTML Cheatsheet

HTML 5 NEW TAG
TAG NOT SUPPORTED IN HTML 5

Tag	Description	Attributes
`<!--...-->`	Define a comment	
`<!DOCTYPE>`	Defines the document type	
`<a>`	Defines a hyperlink	href, hreflang, media, ping, rel, target, type
`<abbr>`	Defines an abbreviation	
`<acronym>`	Used to define an embedded acronyms	
`<address>`	Defines an address element	
`<applet>`	Used to define an embedded applet	
`<area>`	Defines an area inside an image map	alt, coords, href, hreflang, media, ping, rel, shape, target, type
`<article>`	Defines an article	cite, pubdate
`<aside>`	Defines content aside from the page content	
`<audio>`	Defines sound content	autobuffer, autoplay, controls, src
`<b>`	Defines bold text	
`<base>`	Defines a base URL for all the links in a page	href, target
`<basefont>`	Used to define a default font-color, font-size, or font-family for all the document	dir
`<bdo>`	Defines the direction of text display	
`<big>`	Used to make text bigger	
`<blockquote>`	Defines a long quotation	cite
`<body>`	Defines the body element	
` `	Inserts a single line break	
`<button>`	Defines a push button	autofocus, disabled, form, formaction, formenctype, formmethod, formnovalidate, formtarget, name, type, value
`<canvas>`	Defines graphics	height, width
`<caption>`	Defines a table caption	
`<center>`	Used to center align text and content	
`<cite>`	Defines a citation	
`<code>`	Defines computer code text	
`<col>`	Defines attributes for table columns	autobuffer, autoplay, controls, src
`<colgroup>`	Defines groups of table columns	span
`<command>`	Defines a command button	checked, disabled, icon, label, radiogroup, type
`<datalist>`	Defines a dropdown list	
`<dd>`	Defines a definition description	
`<del>`	Defines deleted text	cite, datetime
`<details>`	Defines details of an element	open
`<dialog>`	Defines a dialog (conversation)	
`<dfn>`	Defines a definition term	
`<dir>`	Used to define a directory list	
`<div>`	Defines a section in a document	
`<dl>`	Defines a definition list	
`<dt>`	Defines a definition term	
`<em>`	Defines emphasized text	
`<embed>`	Defines external interactive content or plugin	height, src, type, width
`<fieldset>`	Defines a fieldset	disabled, form, name
`<figure>`	Defines a group of media content, and their caption	
`<font>`	Used to define font face, font size, and font color of text	
`<footer>`	Defines a footer for a section or page	
`<form>`	Defines a form	accept-charset, action, autocomplete, enctype, method, name, novalidate, target
`<frame>`	Used to define one particular window (frame) within a frameset	
`<frameset>`	Used to define a frameset, which organized multiple windows (frames)	
`<h1> to <h6>`	Defines header 1 to header 6	
`<head>`	Defines information about the document	
`<header>`	Defines a header for a section or page	
`<hgroup>`	Defines information about a section in a document	
`<hr>`	Defines a horizontal rule	
`<html>`	Defines an html document	manifest, xmlns
`<i>`	Defines italic text	
`<iframe>`	Defines an inline sub window	height, name, sandbox, seamless, src, width
`<img>`	Defines an image	alt, src, height, ismap, usemap, width
`<input>`	Defines an input field	accept, alt, autocomplete, autofocus, checked, disabled, form, formaction, formenctype, formmethod, formnovalidate, formtarget, height, list, max, maxlength, min, multiple, name, pattern, placeholder, readonly, required, size, src, step, type, value, width
`<ins>`	Defines inserted text	cite, datetime
`<keygen>`	Defines a generated key in a form	autofocus, challenge, disabled, form, keytype, name
`<kbd>`	Defines keyboard text	
`<label>`	Defines an inline sub element	for, form
`<legend>`	Defines a title in a fieldset	
`<li>`	Defines a list item	value
`<link>`	Defines a resource reference	href, hreflang, media, rel, sizes, type
`<map>`	Defines an image map	name
`<mark>`	Defines marked text	
`<menu>`	Defines a menu list	label, type
`<meta>`	Defines meta information	charset, content, http-equiv, name
`<meter>`	Defines measurement within a predefined range	high, low, max, min, optimum, value
`<nav>`	Defines navigation links	
`<noframes>`	Used to display text for browsers that do not handle frames	
`<noscript>`	Defines a noscript section	
`<object>`	Defines an embedded object	data, form, height, name, type, usemap, width
`<ol>`	Defines an ordered list	reversed, start
`<optgroup>`	Defines an option group	label, disabled
`<option>`	Defines an option in a drop-down list	disabled, label, selected, value
`<output>`	Defines some types of output	for, form, name
`<p>`	Defines a paragraph	
`<param>`	Defines a parameter for an object	name, value
`<pre>`	Defines preformatted text	
`<progress>`	Defines progress of a task of any kind	max, value
`<q>`	Defines a short quotation	cite
`<rp>`	Used in ruby annotations to define what to show browsers that to not support the ruby element	
`<rt>`	Defines explanation to ruby annotations	
`<ruby>`	Defines ruby annotations	
`<s>, <strike>`	Used to define strikethrough text	
`<samp>`	Defines sample computer code	
`<script>`	Defines a definition list	async, type charset defer, src
`<section>`	Defines a section	cite
`<select>`	Defines a selectable list	autofocus, disabled, form, multiple, name, size
`<small>`	Defines small text	
`<source>`	Defines media resources	media, src, type
`<span>`	Defines a section in a document	
`<strong>`	Defines strong text	
`<style>`	Defines a style definition	type, media, scoped
`<sub>, <sup>`	Defines sub/super-scripted text	
`<table>`	Defines a table	summary
`<tbody>`	Defines a table body	summary
`<td>`	Defines a table cell	colspan, headers, rowspan
`<textarea>`	Defines a text area	autofocus, cols, disabled, form, maxlength, name, placeholder, readonly, required, rows, wrap
`<tfoot>`, `<thead>`	Defines a table footer / head	
`<th>`	Defines a table header	colspan, headers, rowspan, scope
`<time>`	Defines a date/tim	datetime
`<title>`	Defines the document title	
`<tr>`	Defines a table row	datetime
`<tt>`	Used to define teletype text	
`<u>`	Used to define underlined text	
`<ul>`	Defines an unordered list	
`<var>`	Defines a variable	
`<video>`	Defines a video	autobuffer, autoplay, controls, height, loop, src, width

HTML5 TAG CHEAT SHEET
Created by WebsiteSetup.org

Source: https://websitesetup.org/html5-cheat-sheet/

CSS Cheatsheet

GoSquared — CSS Help Sheet

Syntax

Write styles for any element

selector {property: value;}

External Style Sheet

<link rel="stylesheet" type="text/css" href="style.css" />

Internal Style

<style type="text/css">
selector {property: value}
</style>

Inline Style

<tag style="property: value">

General

class	String preceded by a full stop (.)
ID	String preceded by a hash (#)
div	Formats structure or block of text
span	Inline formatting
color	Foreground colour
cursor	Appearance of the cursor
display	block; inline; list-item; none
overflow	How to handle content that overflows its box. visible; hidden; scroll; auto
visibility	visible; hidden

Font

font-style	italic, normal
font-variant	normal, small-caps
font-weight	bold, normal, lighter, bolder, integer [100-900]
font-size	Size of the font
font-family	Specific font[s] to be used.

Text

letter-spacing	Space between letters
line-height	Vertical space between baselines
text-align	Horizontal alignment
text-decoration	blink, line-through, none, overline, underline
text-indent	First line indentation
text-transform	capitalise, lowercase, uppercase
vertical-align	Vertical alignment
word-spacing	Spacing between words

Margins + Padding

height; width;

margin-top;
margin-right;
margin-bottom;
margin-left;

padding-top;
padding-right;
padding-bottom;
padding-left;

Border

border-width	Width of the border
border-style	dashed; dotted; double; groove; inset; outset; ridge; solid; none;
border-color	Colour of the border

Position

clear	If any floating elements around the element both, left, right, none
float	Floats to a specified side left, right, none
left	The left position of an element auto, length values [pt, in, cm, px]
top	The top position of an element auto, length values [pt, in, cm, px]
position	static, relative, absolute
z-index	Above or below overlapping elements auto, integer [higher numbers on top]

Background

background-color	Colour of background
background-image	Background image url(")
background-repeat	repeat, no-repeat, repeat-x, repeat-y
background-attachment	Background image scrolls with element scroll, fixed
background-position	(x y), top, center, bottom, left, right

List

list-style-type	Type of bullet or numbering in the list disc; circle; square; decimal; lower-roman; upper-roman; lower-alpha; upper-alpha; none
list-style-position	Position of the bullet or number in a list inside; outside
list-style-image	Image to be used as the bullet in the list

Shorthand

background
border
border-bottom
border-left
border-right
border-top
font
list-style
margin
padding

Comments

/* Comments */

Pseudo Selectors

:hover
:active
:focus
:link
:visited
:first-line
:first-letter

Media Types

all
braille
embossed
handheld
print
projection
screen
speech
tty
tv

Units

Length

%
em
pt
px

Keywords

bolder
lighter
larger

Developing for IE6 is a lost cause.

Source: https://www.gosquared.com/blog/panic-over-css2-and-css3-help-sheets-are-here

Advanced Workshop Activity: Your own website

In the advanced workshop, you'll have free reign to create and design your own website with help from the tutors. Create your content locally, then use the information on the next page to upload your new site to the web and show it off to your friends!

Tips to make your life easier…

➢ Remember to give your HTML files ".html" or ".htm" extensions, and your CSS files ".css" extensions!

➢ Name your main/home page "index.html" or "index.htm" in your root directory to ensure it's the default page for new visitors to your website.

➢ Using relative links and references to files within your website structure ensures that they won't break if you later rehost your content at a different web address. Remember to use forward slashes ("/") when referencing files in different folders, not backslashes like in Windows ("\").

➢ Separating your stylesheets from your HTML files will make it a whole lot easier to change the look and feel of your website later!

➢ Just like on your home computer, it's much easier to keep track of all your files and images if you keep a neat directory structure on your webspace. Best practices: don't use spaces in file or folder names, and organise non-HTML files into folders by type (eg. images, css).

Not sure what to put on your site? Here are some ideas:

❖ Create a site about one of your interests or hobbies
❖ Set up your own blog, designed from scratch
❖ Design a gallery site for photos you've taken or images you like
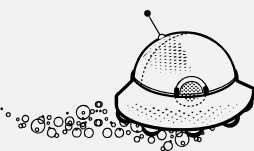❖ Redesign the CSS Zen Garden (see page 7)
❖ Create an online resume for yourself

REMEMBER! Content on the web is public and anyone with the link can see it, so think twice and check with your parents before posting any personally identifying information on your website!
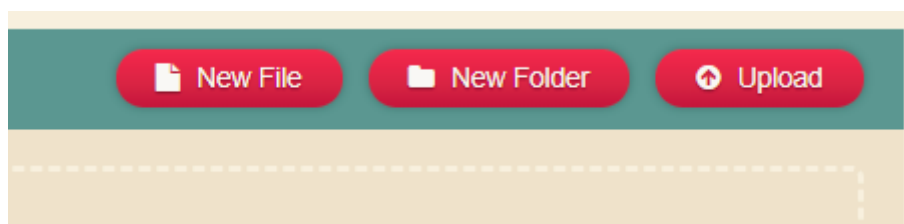
Uploading to the Internet

Neocities is a completely free web host where anyone can create a site and make it available to the world. We're going to use Neocities to upload and host our files.

➢ Sign up to Neocities at **neocities.org**. Your website address is public, so you might not want to use your full name – maybe use your first name and last initial instead, or something describing what your site is about.

➢ Once you've signed up, picked the free plan and confirmed your email address, head to the dashboard at **https://neocities.org/dashboard**

➢ Feel free to delete the starter files Neocities gives you – you have your own content to upload!

Remember to follow common sense about what you upload to your website! It's public, so don't share anything you wouldn't want other people to see.

Uploading files is super easy with Neocities – just drag and drop them from your desktop or Windows Explorer over the top of the Neocities window, or click the red "Upload" button to pick a file from your computer:

Make sure you upload all the files you need – don't forget your CSS files! You can create and edit new files right in your browser with the "New File" button and built-in editor too, so you can work on your website from anywhere.
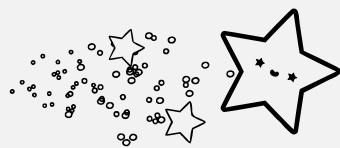
Access your website at **yourname.neocities.org**! Share this address with family and friends to show them what you've been doing at Summer School!

Advanced Workshop Activity: CSS in the Garden

As part of or in addition to making your own website, take a stroll in the CSS Zen Garden: **www.csszengarden.com**

Every design featured in the Zen Garden uses exactly the same HTML – only the CSS is different! It's a beautiful and varied display of the design possibilities using CSS as well as a fantastic example of the concept of *separation of presentation and content*. While the Garden is no longer being updated, it remains online to inspire and challenge web designers.

Once you've finished the Grok course on HTML and CSS, it's time to put what you've learned to the test. Download the example HTML file from CSS Zen Garden (and, if you choose, an example CSS file – but you might want to start from scratch) and try some of your new techniques to style a page without changing the HTML. This is just like the *inTRIPid* activity at the end of your Grok course – but now you have the freedom to design however you want!

> Feel free to restyle the **inTRIPid** page instead, or cook up something of your own!

Where might I start?

➤ Try floating some divs in different ways and see how they move around or how you can position them

➤ Change some colours or find some images you like online and use them in the design

➤ Pick some Zen Garden stylesheets you like, download their source code and copy some properties or look them up in the online CSS reference. Try to figure out *what* they're doing and *how*!

Where do I go from here?

Make a backend

At this camp you've learnt some Python skills, so why not combine those with your new frontend HTML/CSS knowledge and build a backend for a website? A framework like **Django** or **web2py** lets you build dynamic websites rapidly without having to learn database query languages. Try the tutorials! (Note that you won't be able to run these on Neocities, but you can set it up to work on your home computer and buy a plan from a fully-featured web host like VentraIP **www.ventraip.com.au** when you're ready to go live!)

> ➢ **www.djangoproject.com**
> ➢ **www.web2py.com**

Seek external validation

It's always good to check your code for unintended errors – this will make it more likely to display the same way in different browsers, and it's just good coding practice! Luckily, the W3C provides official validators for HTML and CSS (and some other content types too!) that will help you track down errors.
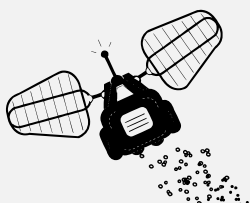
> ➢ **validator.w3.org**
> ➢ **jigsaw.w3.org/css-validator/**

Add interactivity with JavaScript

JavaScript code will run right in your browser and, while it's not limited to use in websites, you can attach it to your HTML elements to have your page interact with the user. There are plenty of guides online - try the tutorial at **htmldog.com/guides/javascript/**. We'll check out JavaScript at winter camp.

Extra references

> ➢ **dev.w3.org/html5/html-author/** - official HTML 5 reference
> ➢ **developer.mozilla.org/docs/Web/HTML** - Mozilla HTML reference/guide
> ➢ **developer.mozilla.org/docs/Web/CSS** - Mozilla CSS reference/guide
> ➢ **www.edx.org/school/w3cx** - Free online HTML and CSS courses
> from the W3C, the web's international standards organisation

You can make as many Neocities sites as you like. What can you come up with? You might also like to ask your IT teacher if your school provides space for student websites!