CSE 154: Web Programming

Exam 1 “Cheat Sheet”

Note that this is not a comprehensive cheat sheet for HTML/CSS/JS, but provides a quick reference for common tags, terminology, styles, properties, etc. you may find helpful during a CSE 154 exam.

HTML

Tags Used in the `<head>` Section

Tag	Description
`<title>` **text** `</title>`	title shown on page tab
`<meta` **attribute="value"** `... />`	page metadata
`<link href="`**filepath**`" rel="stylesheet" />`	links to a CSS style sheet
`<script src="`**filepath**`"></script>`	link to JavaScript code
`<!--` **comments** `-->`	comment (can appear in `head` or `body`)

Tags Used in the `<body>` Section

Tag	Display	Description
`<p>`**text** `</p>`	Block	paragraph

`<h1>`**text** `</h1>`
`<h2>`**text** `</h2>`
…
`<h6>`**text** `</h6>`
	Block	(h1 for largest to h6 for smallest)

`<hr />`	Block	horizontal rule (line)
`<br />`	Inline	line break
`<a href="`**url**`">`**text** `</a>`	Block	anchor (link)
`<img src="`**url**`" alt="`**description**`" />`	Inline-block	image
`<em>`**text**`</em>`	Inline	emphasis (italic)
`<strong>`**text** `</strong>`	Inline	strong emphasis (bold)

`<ol>`
  `<li>`**text** `</li>`
  `<li>`**text** `</li>`
  `<li>`
    `<ul>`
      `<li>`**nested item text**`</li>`
      `<li>`**nested item text**`</li>`
    `</ul>`
  `</li>`
`</ol>`
	Block	ordered (`ol`) and unordered (`ul`) list;
		list item (`li`)

Tags Used in the `<body>` Section (Continued)

Tag	Display	Description
`<blockquote>` `<p>`**text**`</p> ...` `</blockquote>`	Block	block-level quotation
`<q>`**text**`</q>`	Inline	inline-level quotation
`<code>`**text**`</code>`	Inline	computer code (monospace)
`<pre>`**text**`</pre>`	Inline	pre-formatted **text** (preserves whitespace)
`<table>` `<caption>`**text**`</caption>` `<tr>` `<th>`**heading 1**`</th>` `<th>`**heading 2**`</th>` `</tr>` `<tr>` `<td> cell 1 </td>` `<td> cell 2 </td>` `</tr>` `...` `</table>`	Block	table of data (`table`) description of table (`caption`) table row (`tr`) table heading cell (`hr`) normal table cell (`td`)
`<div> ... </div>`	Block	block-level section of a page
`<span> ... </span>`	Inline	inline-level section of a page

HTML5 Semantic Grouping Tags (all block elements)

Tag	Description
`<header>`	Container for a header of a document
`<main>`	Specifies the main content of a document. The content inside should be unique to the document and not contain content that is repeated across pages (e.g., sidebars, nav links, search bars, etc.)
`<footer>`	Container for a footer of a document
`<article>`	A standalone piece of content (e.g., entire blog post including title, author, etc.)
`<section>`	A piece of content that is part of another (e.g., a chapter section of a reading)
`<aside>`	Defines some content aside from the content it is placed in (e.g., a sidebar in an article)
`<nav>`	Defines content in a navigation bar

HTML Input Tags

Tag	Display	Description
`<button>`**content**`</button>`	Inline	button element
`<input type="`**type**`" name="`**name**`" />`	Inline	form element input tag type can be `text, number, checkbox, radio, file, etc.`

`<textarea rows="`**num**`" cols="`**num**`">`
initial text
`</textarea>`
	Inline	multi-line **text** input box

`<label>... </label>`
`<label for="input-id">`**text**`</label>`
	Inline	clickable **text** label around a form control or linked to a form control using the control's id in `for` attribute

`<select>`
`<option>`**text** `</option>`
`...`
`</select>`
	Inline	drop-down selection box (`select`);
		each option within the box (`option`);

HTML Entities Reference

Result	Description	Entity Name
	non-breaking space	`&nbsp;`
<	less than	`&lt;`
@	at symbol	`&commat;`
>	greater than	`&gt;`
&	ampersand	`&amp;`
©	copyright	`&copy;`

CSS

Selector Types

Name	Description	Example(s)
Element	Any element of a given type	`h1 { ... }`
Grouping	Multiple elements of different types	`h1, h2, li.bordered, { ... }`
Class	Elements with the given class name	`.example { ... }`
Id	Single element with the given id	`#example { ... }`
Descendant	Elements that are children at any level of another specified element	`section header { ... }`
Child	Elements that are direct children of another specified element	`section > header { ... }`
Attribute	Elements that have the specified attribute	`input[disabled]` - inputs that have the `disabled` (boolean) attribute `input[name='test']`- inputs that have a name 'test'

Color Values

Value	Description
`colorname`	Standard name of color, such as red, blue, purple, etc.
`rgb(redval, greenval, blueval)`	Example: red = rgb(255, 0, 0) or red = rgb(100%, 0, 0)
`#RRGGBB`	Example: red = #FF0000, green = #00FF00, white = #FFFFFF

For the following property and value tables, anything *emphasized* represents values that should be replaced with specific units (e.g., *`length`* should be replaced with a `px`, `pt`, or `em` for many properties, and *`color`* should be replaced with a valid color value such as a hex or rgb code).

A use of | refers to separation of possible values (where you cannot provide two of these possible values for one property) and [value value value] refers to a grouping of possible values that can optionally be used together (e.g., *`[width style color]`* for the `border` shorthand).

Font and Text Styles

Property	Values
`font-style`	`normal | italic | oblique | inherit`
`font-family`	*`fontname`*
`font-size`	*`length`* `| %`
`font-weight`	`normal | bold | inherit`
`text-align`	`left | right | center | justify`
`text-decoration`	`none | [underline overline line-through]`
`text-transform`	`none | capitalize | uppercase | lowercase`

Background Styles

Property	Values
`background-color`	*`color`* `| transparent`
`background-image`	*`url`* `| none`
`background-size`	*`length`* `| % | auto | cover | contain`
`background-repeat`	`repeat | repeat-x | repeat-y | no-repeat`

Note for **margin**, **padding**, and **border** (Box Model): Replace '*' with any side of the box model (top, right, left, bottom) for the desired effect. Example style: '`border: 2px solid red`' applies a solid red border with a 2px width to all four sides, while '`border-left: 2px solid red`' only applies to the left border.

Border Styles

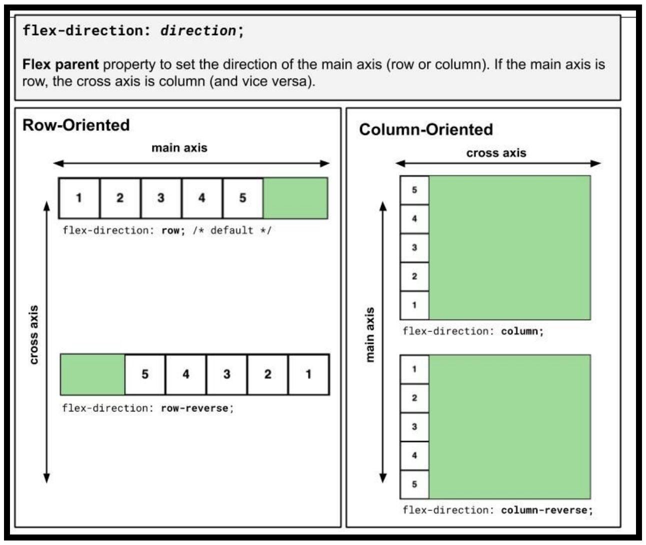
Property	Values
Shorthands for `border`, `border-*`	*`[width style color]`*
`border-width`, `border-*-width`	`thin | medium | thick |` *`length`*
`border-style`, `border-*-style`	`none | hidden | dotted | dashed | solid`
`border-color`, `border-*-color`	*`color`*
`border-radius`	*`length`*

Box Model/Layout

Property	Values
`height, width`	`auto` | *`length`* | `%`
`min-height, max-height` `min-width, max-width`	`none` | *`length`* | `%`
`margin, margin-*`	`auto` | *`length`* | `%`
`padding, padding-*`	*`length`* | `%`
`display`	`none` | `inline` | `block` | `inline-block` | `flex`
`float`	`left` | `right` | `none`
`overflow, overflow-x, overflow-y`	`visible` | `hidden` | `scroll` | `auto`
`clear`	`left` | `right` | `both` | `none`
`position`	`absolute` | `relative` | `static` | `fixed` | `sticky`

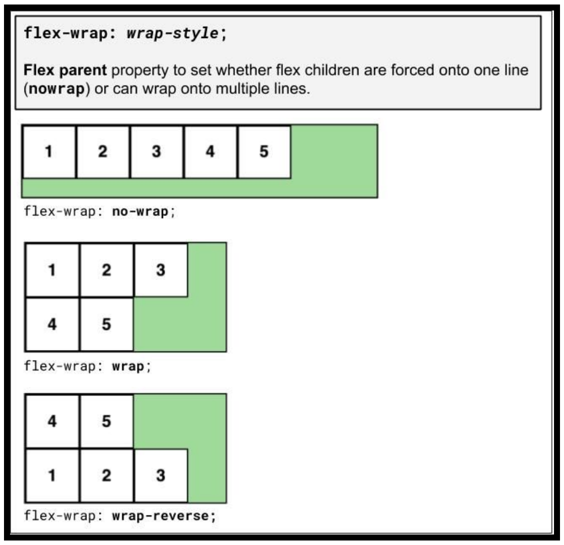
Flex Layout

Below are diagrams of the most common flex properties for flex containers/items.



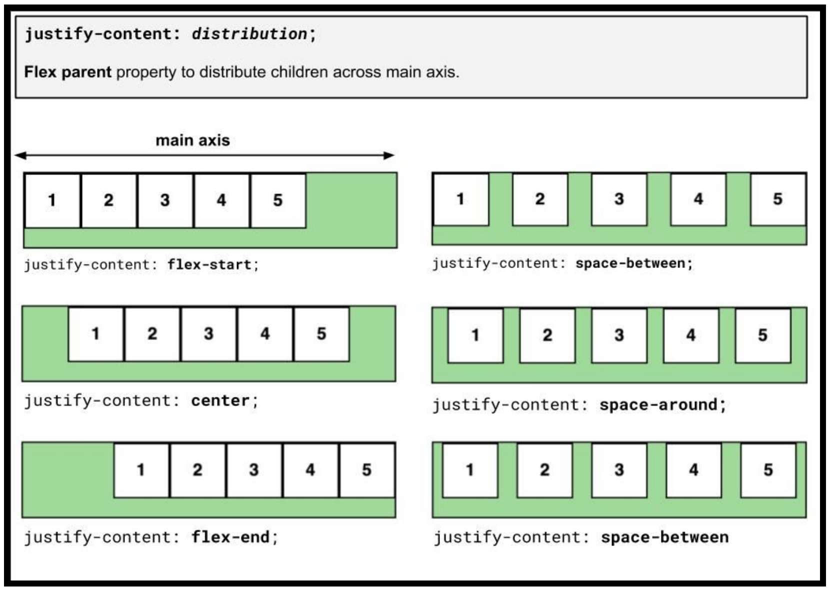
`flex-direction: `***`direction`***`;`

Flex parent property to set the direction of the main axis (row or column). If the main axis is row, the cross axis is column (and vice versa).



`flex-wrap: `***`wrap-style`***`;`

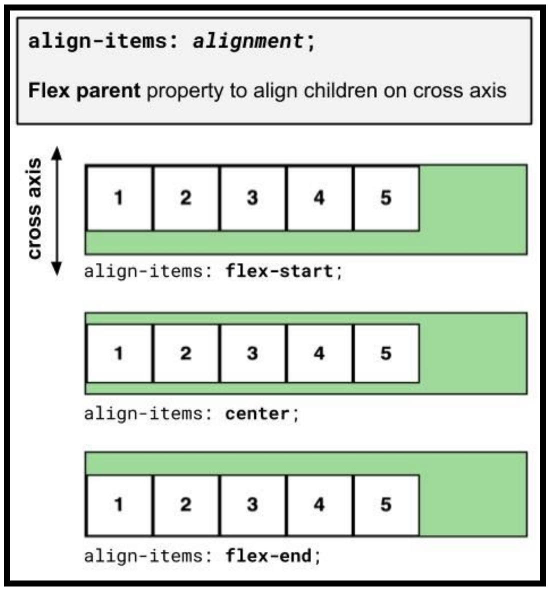
Flex parent property to set whether flex children are forced onto one line (**`nowrap`**) or can wrap onto multiple lines.



`justify-content: `***`distribution`***`;`

Flex parent property to distribute children across main axis.

main axis
1 2 3 4 5
justify-content: flex-start;
1 2 3 4 5
justify-content: space-between;
1 2 3 4 5
justify-content: center;
1 2 3 4 5
justify-content: space-around;
1 2 3 4 5
justify-content: flex-end;
1 2 3 4 5
justify-content: space-between



`align-items: `***`alignment`***`;`

Flex parent property to align children on cross axis

JavaScript

`window` Methods and Properties

Method/Property	**Description**
`document`	Returns a reference to the document contained in the window
`getComputedStyle(element)`	Returns an object that reports the values of all CSS properties of an element after applying active stylesheets and resolving any basic computation those values may contain

`document` Methods

Method/Property	**Description**
`getElementById(id)`	Returns a DOM object whose id property matches the specified string. If no matches are found, null is returned.
`querySelector(sel)`	Returns the first DOM element that matches the specified selector, or group of selectors. If no matches are found, null is returned.
`querySelectorAll(sel)`	Returns a list of the document's elements that match the specified group of selectors. If no matches are found, null is returned.
`createElement(tagName)`	Creates and returns an Element node with the given tag name

DOM Element Methods and Properties

Method/Property	**Description**
`el.id`	Sets or returns the value of the id attribute of an element
`el.getAttribute(attr)`	Returns the specified attribute value `attr` of `el`
`el.textContent`	Sets or returns the text content of the specified node
`el.innerHTML`	Sets or returns the HTML content of an element
`el.classList`	Returns the class name(s) of `el`
`el.className`	Sets or returns the value of the class attribute of `el`
`el.addEventListener(event, fn)`	Attaches an event handler function `fn` to the specified element `el` to listen to `event`
`el.removeEventListener(event, fn)`	Removes the event handler `fn` to the specified `el` listening to `event`
`el.children`	Returns a collection of the child elements of `el`
`el.parentNode`	Returns the parent node of `el`
`el.appendChild(child)`	Adds a new child node to `el` as the last child node
`el.insertBefore(newNode, refNode)`	Adds newNode to parent `el` before `el`'s child `refNode` position
`el.removeChild(child)`	Removes a child node from a parent element

Accessing DOM Element Attributes

Recall that if you have an HTML element on your page that has attributes, you can set those properties through JavaScript as well. For instance:

```
<img id="dog-tag" src="img/doggie.jpg" alt="My Cute Dog" />
```

Your could do the following in your JavaScript code (using the `id` alias for `document.getElementById`):

```
id("dogtag").alt = "My really cute dog";
```

Example DOM Element attributes include (other than `src`, and `alt` above) are:

Property	Description
`disabled`	Whether or not this DOM element is disabled on the page (boolean)
`value`	The current value of form elements (`input`, `textarea`, `checkbox` `radio`, `select`, etc.)
`name`	The value of the name attribute of a form element

DOM Element `.classList` Methods and Properties

Method/Property	Description
`add(class)`	Adds specified class values. These values are ignored if they already exist in the list
`remove(class)`	Removes the specified class value if it exists
`toggle(class)`	Toggles the listed class value. If the class exists, then removes it and returns false, otherwise adds it to the list and returns true
`contains(class)`	Returns true if the specified class value exists in the classList

Common Event Types

`load`	`mouseout`	`mouseup`	`keydown`	`change`
`click`	`mouseover`	`mouseenter`	`keyup`	`error`
`dblclick`	`mousedown`	`submit`	`select`	`success`

JavaScript Methods Useful with JSON/Objects

`{ key : value, key : value, … }`

Function	Description
`parse(string)`	Returns the given string of JSON data as the equivalent JavaScript object
`stringify(object)`	Returns the given object as a string of JSON data
`Object.keys(data)`	Returns an array of keys of the given object

Other handy JavaScript Methods

Function	Description
`parseInt(value)`	Returns the integer representation of the given value, if it starts with a Number-like type Examples: `parseInt("12px")` evaluates to 12, `parseInt(10.5)` evaluates to 10
`console.log(data)`	Outputs the `data` to the JavaScript console

JavaScript `string` Methods and Properties

Method/Property	Description
`length`	Returns the length of a string
`charAt(index)`	Returns the character at the specified index
`indexOf(str)`	Returns the position of the first occurrence of a specified value in a string (-1 if not found)
`split(delimiter)`	Splits a string into an array of substrings based on delimiter
`substring(start, end)`	Extracts the characters from a string between two specified indices (start is inclusive, end is exclusive)
`trim()`	Removes whitespace from both ends of a string
`toLowerCase()`	Returns a lowercase version of a string
`toUpperCase()`	Returns an uppercase version of a string

JavaScript Array Methods and Properties

Method/Property	Description
`length`	Sets or returns the number of elements in an array
`push(el)`	Adds new elements to the end of an array and returns the new length
`pop()`	Removes and returns the last element of an array
`unshift(el)`	Adds new elements to the beginning of an array and returns the new length
`shift()`	Removes and returns the first element in an array
`sort()`	Sorts the elements of an array
`indexOf(el)`	Returns the index of the element in the array, or -1 if not found

JavaScript Timer Functions

Method	Description
`setTimeout(fn, ms)`	Executes a function `fn` after a delay of `ms` milliseconds. Returns a Number value representing the ID of the timeout being set.
`setInterval(fn, ms)`	Executes a function `fn` at every given time-interval (in milliseconds). Returns a Number value representing the ID of the interval being set.
`clearTimeout(timerId)`	Stops the execution of the delay timer specified by timerId
`clearInterval(timerId)`	Stops the execution of the interval timer specified by timerId

JavaScript Math Functions

Method	Description
`Math.random()`	Returns a double between 0 (inclusive) and 1 (exclusive)
`Math.abs(n)`	Returns the absolute value of `n`
`Math.min(a, b, ...)`	Returns the smallest of 0 or more numbers
`Math.max(a, b, ...)`	Returns the largest of 0 or more numbers
`Math.round(n)`	Returns the value of `n` rounded to the nearest integer
`Math.ceil(n)`	Returns the smallest integer greater than or equal to `n`
`Math.floor(n)`	Returns the largest integer less than or equal to `n`

The Module Pattern

Whenever writing JavaScript, you should use the module pattern, wrapping the content of the code (`window load` event handler and other functions) in an anonymous function. Below is a template for reference:

```
“use strict”;
(function() {

  // any module-globals (limit the use of these when possible)
  window.addEventListener(“load”, init);

  function init() {
    ...
  }
  // other functions
})();
```

Helper Alias Functions

You may use any of the following alias functions in your exam without defining them:

```
function id(idName) {
  return document.getElementById(idName);
}

function qs(selector) {
  return document.querySelector(selector);
}

function qsa(selector) {
  return document.querySelectorAll(selector);
}

function gen(tagName) {
  return document.createElement(tagName);
}
```