

## Exam "Cheat Sheet"

### HTML

#### Tags Used in the <head> Section

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

#### Tags Used in the <body> Section

Tag	Display	Description
<code>&lt;p&gt; text &lt;/p&gt;</code>	Block	paragraph
<code>&lt;h1&gt; text &lt;/h1&gt;</code> <code>&lt;h2&gt; text &lt;/h2&gt;</code> ... <code>&lt;h6&gt; text &lt;/h6&gt;</code>	Block	(h1 for largest to h6 for smallest)
<code>&lt;hr /&gt;</code>	Block	horizontal rule (line)
<code>&lt;br /&gt;</code>	Block	line break
<code>&lt;a href="url"&gt; text &lt;/a&gt;</code>	Block	anchor (link)
<code>&lt;img src="url" alt="description" /&gt;</code>	Inline-Block	image
<code>&lt;em&gt; text &lt;/em&gt;</code>	Inline	emphasis (italic)
<code>&lt;strong&gt; text &lt;/strong&gt;</code>	Inline	strong emphasis (bold)
<code>&lt;ol&gt;</code> <code>&lt;li&gt; text &lt;/li&gt;</code> <code>&lt;li&gt; text &lt;/li&gt;</code> <code>&lt;li&gt;</code> <code>&lt;ul&gt;</code> <code>&lt;li&gt; nested item &lt;/li&gt;</code> <code>&lt;li&gt; nested item &lt;/li&gt;</code> <code>&lt;/ul&gt;</code> <code>&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	Block	ordered (ol) and unordered (ul) list; list item (li)

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

Tag	Display	Description
<code>&lt;dl&gt;</code> <code>&lt;dt&gt; term 1 &lt;/dt&gt;</code> <code>&lt;dd&gt; description 1 &lt;/dd&gt;</code> <code>&lt;dt&gt; term 2 &lt;/dt&gt;</code> <code>&lt;dd&gt; description 2 &lt;/dd&gt;</code> <code>&lt;/dl&gt;</code>	Block	definition list (dl); term (dt), and its description (dd)
<code>&lt;blockquote&gt;</code> <code>&lt;p&gt; text &lt;/p&gt; ...</code> <code>&lt;/blockquote&gt;</code>	Block	block-level quotation
<code>&lt;q&gt; text &lt;/q&gt;</code>	Inline	inline-level quotation
<code>&lt;code&gt; text &lt;/code&gt;</code>	Inline	computer code (monospace)
<code>&lt;pre&gt; text &lt;/pre&gt;</code>	Inline	pre-formatted text (preserves whitespace)
<code>&lt;table&gt;</code> <code>&lt;caption&gt; text &lt;/caption&gt;</code> <code>&lt;tr&gt;</code> <code>&lt;th&gt; heading 1 &lt;/th&gt;</code> <code>&lt;th&gt; heading 2 &lt;/th&gt;</code> <code>&lt;/tr&gt;</code> <code>&lt;tr&gt;</code> <code>&lt;td&gt; cell 1 &lt;/td&gt;</code> <code>&lt;td&gt; cell 2 &lt;/td&gt;</code> <code>&lt;/tr&gt;</code> ... <code>&lt;/table&gt;</code>	Block	table of data (table) description of table (caption) table row (tr) table heading cell (th) normal table cell (td)
<code>&lt;div&gt; ... &lt;/div&gt;</code>	Block	block-level section of a page
<code>&lt;span&gt; ... &lt;/span&gt;</code>	Inline	inline-level section of a page

## HTML5 Semantic Grouping Tags

Tag	Display	Description
<code>&lt;header&gt;</code>	Block	Container for a header of a document
<code>&lt;footer&gt;</code>	Block	Container for a footer of a document
<code>&lt;article&gt;</code>	Block	A standalone piece of content (e.g., entire blog post including title, author, etc.)
<code>&lt;section&gt;</code>	Block	A piece of content that is part of another (e.g., a chapter section of a reading)
<code>&lt;aside&gt;</code>	Block	Defines some content aside from the content it is placed in (e.g., a sidebar in an article)
<code>&lt;main&gt;</code>	Block	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.)

## HTML Input Tags

Note that input tags are inline tags.

Tag	Description
<code>&lt;button type="type"&gt;</code> <b>content</b> <code>&lt;/button&gt;</code>	clickable button type can be submit, reset, button
<code>&lt;input type="type" name="name"&gt;</code> <b>content</b> <code>&lt;/input&gt;</code>	form input tag type can be text, number, submit, reset, checkbox, radio, file, etc.
<code>&lt;textarea rows="num" cols="num"&gt;</code> <b>initial text</b> <code>&lt;/textarea&gt;</code>	multi-line <b>text</b> input box
<code>&lt;label&gt; text &lt;/label&gt;</code>	clickable <b>text</b> label around a form control
<code>&lt;select&gt;</code> <code>&lt;option&gt; text &lt;/option&gt;</code> <code>&lt;option&gt;</code> <code>&lt;optgroup label="text"&gt;</code> <code>&lt;option&gt; text &lt;/option&gt;</code> <code>&lt;option&gt; text &lt;/option&gt;</code> <code>&lt;/optgroup&gt;</code> ... <code>&lt;/select&gt;</code>	drop-down selection box (select); each option within the box (option); a labeled group of options (optgroup);
<code>&lt;fieldset&gt;</code> <code>&lt;legend&gt; text &lt;/legend&gt;</code> content <code>&lt;/fieldset&gt;</code>	a grouped set of form fields with a legend

## HTML Entities Reference

Result	Description	Entity Name
	non-breaking space	<code>&amp;nbsp;</code>
<code>&lt;</code>	less than	<code>&amp;lt;</code>
<code>&gt;</code>	greater than	<code>&amp;gt;</code>
<code>&amp;</code>	ampersand	<code>&amp;amp;</code>
<code>©</code>	copyright	<code>&amp;copy;</code>

## CSS

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., [*h-shadow v-shadow blur spread color*] for box-shadow).

### Selector Types

Name	Description	Example
Universal	Any element	<code>* { font: 10pt Arial; }</code>
Element	Any element of a given type	<code>h1 { text-decoration: underline; }</code>
Grouping	Multiple elements of different types	<code>h1, h2, h3 { color: purple; }</code>
Class	Elements with the given class name	<code>.example { text-decoration: underline; }</code>
Id	Single element with the given id	<code>#example { text-decoration: overline; }</code>
Descendant	Elements that are children at any level of another specified element	<code>#example h1 { text-decoration: underline; }</code>
Child	Elements that are direct children of another specified element	<code>#example &gt; p { font-weight: bold; }</code>
Attribute	Elements that have the specified attribute	<code>input[selected]</code> - inputs that have the selected attribute  <code>input[name='test']</code> - inputs that have a name 'test'

### Background Styles

Property	Values
background-attachment	scroll   fixed
background-color	<i>color</i>   transparent
background-image	<i>url</i>   none
background-origin	border-box   padding-box   content-box
background-position	top left   top center   top right   center left   center center   center right   bottom left   bottom center   bottom right [ <i>x-% y-%</i> ]   [ <i>x-pos y-pos</i> ]
background-size	<i>length</i>   %   auto   cover   contain
background-repeat	repeat   repeat-x   repeat-y   no-repeat
background-attachment	scroll   fixed

## Border Styles

Note: Replace '\*' with any side of the border (top, right, left, bottom) for the desired effect.

Example style: 'border: 2px solid red' applies a solid red border with a width of 2px to all four sides of the element, while 'border-left: 2px solid red' only applies that border to the left border'.

Property	Values
border, border-* (shorthand)	border-width, border-*-width border-style, border-*-style border-color, border-*-color
border-width, border-*-width	thin   medium   thick   length
border-style, border-*-style	none   hidden   dotted   dashed   solid   double   groove   rigid   inset   outset
border-color, border-*-color	<i>color</i>
box-shadow	none   inset   [ <i>h-shadow v-shadow blur spread color</i> ]
border-radius	<i>length</i>

## Font and Text Styles

Property	Values
font-style	normal   italic   oblique   inherit
font-family	<i>fontname</i>
font-size	<i>length</i>   %
font-weight	normal   bold   inherit
text-align	left   right   center   justify
text-decoration	none   [underline overline line-through blink]
text-shadow	none   [ <i>color length</i> ]
letter-spacing, word-spacing	normal   <i>length</i>   %
text-indent	<i>length</i>   %
text-transform	none   capitalize   uppercase   lowercase
list-style-type	none   asterisks   box   check   diamond   disc   hyphen   square   decimal   lower-roman   upper-roman   lower-alpha   upper-alpha   lower-greek   upper-greek   lower-latin   upper-latin   footnotes
list-style-image	none   <i>url</i>

## Color Values

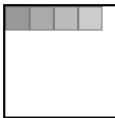
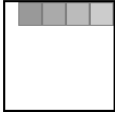
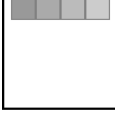
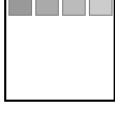
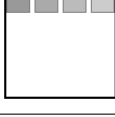
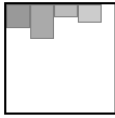
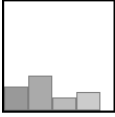
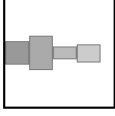

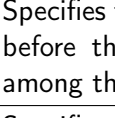
Value	Description
colorname	standard name of color, such as red, blue, purple, etc.
rgb(redvalue, greenvalue, bluevalue)	Example: red = rgb(255, 0, 0) or red = rgb(100%, 0, 0)
#RRGGBB	Example: red = #FF0000

## Box Model

Property	Values
float	left   right   none
height, width	auto   <i>length</i>   %
min-height, max-height min-width, max-width	none   <i>length</i>   %
margin, margin-*	auto   <i>length</i>   %
padding, padding-*	<i>length</i>   %
display	none   inline   block   inline-block   flex   list-item   compact   table   inline-table
overflow, overflow-x, overflow-y	visible   hidden   scroll   auto   no-display   no-content
clear	left   right   both   none

## Flex Box

(on next page)

Property	Values	Element Type	Description
display	flex	Flex Container	Sets all children to become 'flex-items'
justify-content	flex-start  flex-end  center  space-around  space-between	Flex container	Indicates how to position the flex-items in the parent container     
flex-direction	row   row-reverse   column   column-reverse	Flex container	Indicates if the container flows horizontally (row) or vertically (column)
align-items	stretch (default)  flex-start  flex-end  center  baseline	Flex container	Indicates how to space the items inside the container along the cross axis     
flex-basis	auto (default)   length   %	Both	Specifies the default size of an element before the extra space is distributed among the flex-items
order	number	Flex item	Specifies the order in which the element appears in the flex container (by default, flex items are laid out in the source order)
align-self	flex-end   flex-start   center   baseline   stretch (default)	Flex item	Indicates where to place this specific item along the cross axis

# JavaScript

## Window Methods and Properties

Method/Property	Description
<code>document</code>	Returns a reference to the document contained in the window
<code>getComputedStyle(element)</code>	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 and Properties

Method/Property	Description
<code>getElementById(id)</code>	Returns a DOM object whose id property matches the specified string
<code>getElementsByClassName(cn)</code>	Returns an array-like object of all elements which have all of the given class names
<code>getElementsByTagName(name)</code>	Returns an collection of all elements which have all of the given name
<code>querySelector(sel)</code>	Returns the first DOM element that matches the specified selector, or group of selectors. If no matches are found, null is returned
<code>querySelectorAll(sel)</code>	Returns a list of the document's elements that match the specified group of selectors.
<code>getElementsByTagName(name)</code>	Returns a NodeList containing all elements with the specified tag name
<code>createElement(eltType)</code>	Creates and returns an Element node
<code>createTextNode</code>	Creates and returns a Text node

## DOM Element Methods and Properties

Method/Property	Description
<code>children</code>	Returns a collection of an element's child elements
<code>parentNode</code>	Returns the parent node of an element
<code>classList</code>	Returns the class name(s) of an element
<code>className</code>	Sets or returns the value of the class attribute of an element
<code>appendChild(child)</code>	Adds a new child node, to an element as the last child node
<code>addEventListener(event, fn)</code>	Attaches an event handler to the specified element
<code>getAttribute(attr)</code>	Returns the specified attribute value attr of an element node
<code>innerHTML</code>	Sets or returns the content of an element
<code>innerText</code>	Sets or returns the text content of the specified node
<code>id</code>	Sets or returns the value of the id attribute of an element
<code>removeChild(child)</code>	Removes a child node from an element



## Other DOM Element Properties

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 if your

```

```

You could do the following in your JavaScript code (using the \$ alias for document.getElementById):

```
$("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
value	The value of the value attribute of a text field
name	The value of the name attribute of a form element
href	The href for a link or a tag

## 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
toggle(class)	Toggles the listed class value. If the class exists, then removes it and returns false, if it did not exist in the list add it and return true
contains(class)	Returns true if the specified class value exists in the classList for this element

## Event Object Methods and Properties

Method/Property	Description
target	Returns the element that triggered the event
type	Returns the name of the event
offsetX	Returns the horizontal coordinate of the mouse pointer, relative to the DOM element clicked
offsetY	Returns the vertical coordinate of the mouse pointer, relative to the DOM element clicked
stopPropagation	Prevents further propagation of an event during event flow

## Event Types

click	mousemove	keydown	change
dblclick	mouseout	error	focus
mouseenter	mouseover	success	submit
mouseleave	mouseup	load	select
mousedown	keyup	unload	resize

## JavaScript JSON Methods

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

## Other handy JavaScript Methods

Function	Description
parseInt(string, radix)	function parses a string argument and returns an integer of the specified radix (the base in mathematical numeral systems)
console.log(string)	Writes the string to the JavaScript console

## JavaScript Array Methods and Properties

Method/Property	Description
length	Sets or returns the number of elements in an array
push(e1)	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(e1)	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
slice(start, end)	Selects a part of an array and returns the new array
join()	Joins all elements of an array into a string
concat(list2, ...)	Joins two or more arrays and returns a copy of the joined arrays
toString()	Converts an array to a string and returns the result
indexOf(e1)	Returns the index of the element in the array, or -1 if not found

## 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(string)	Returns the position of the first found occurrence of a specified value in a string
split(delimiter)	Splits a string into an array of substrings
substring(start, end)	Extracts the characters from a string between two specified indices
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
concat(str2, ...)	Joins two or more strings and returns a new joined string

## JavaScript Timer Functions

Method	Description
setTimeout(fn, ms)	Executes a function after waiting a specified number of milliseconds. Returns a value representing the ID of the timeout being set.
setInterval(fn, ms)	Repeats a given function at every given time-interval. Returns a value representing the ID of the interval being set.
clearTimeout(id)	Stops the execution of the function specified by id
clearInterval(id)	Stops the execution of the functions specified by id

## 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
Math.pow(n, e)	Returns the base n to the exponent e power, that is, $n^e$
Math.sqrt(n)	Returns the square root of n (NaN if n is negative)

## The Module Pattern

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

```
(function() {  
  // any module-globals (limit the use of these when possible)  
  
  window.onload = function() {  
    ...  
  };  
  
  // other functions  
})();
```

## Helper Alias Functions

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

```
function $(id) {  
  return document.getElementById(id);  
}  
  
function qs(selector) {  
  return document.querySelector(selector);  
}  
  
function qsa(selector) {  
  return document.querySelectorAll(selector);  
}
```

## Javascript Ajax Fetch Skeleton

//you can assume `checkStatus` is already included

```
function checkStatus(response) {  
  if (response.status >= 200 && response.status < 300) {  
    return response.text();  
  } else {  
    return Promise.reject(new Error(response.status + ": " + response.statusText));  
  }  
}  
  
function callAjax(){  
  let url = ..... // put url string here  
  fetch(url) // don't worry about cloud9 credentials  
    .then(checkStatus)  
    .then(JSON.parse) //optional line for processing json  
    .then(function(responseJSON) {  
      //success: do something with the responseJSON  
    })  
    .catch(function(error) {  
      //error: do something with error  
    });  
}
```