**Core 2 Interaction Lab Test Guide**

**HTML structure and elements**

* + - **What is HyperText Markup Language?**

basic language to create website

* + - **Main sections / elements used in an HTML document**

Html/Head/body/style/script/link…

* + - **Parent vs children**

里面的是children，包在外面的是parent

**Embedded content**

* + - **Examples of content that can be embedded**

Image/ audio/ iframes/ viedo

<img src="example.jpg" alt="Description of the image">

* + - **How to embed content**

“src” attribute with the URL of the image file. alternate text using the “alt” attribute

<img src="example.jpg" alt="Description of the image">

**Hyperlink**

* + - **Ways of using the hyperlink (<a> element)**

Linking to Another Webpage

Linking to Specific Section within the Same Page

Linking to an Anchor Point on Another Page

Linking with JavaScript Actions

Linking to Email Addresses/ Phone number

**CSS intro & syntax**

* + - **What is Cascading Style Sheets and how can it be used?**

control the layout, formatting, and appearance of HTML elements on a webpage.

Styling HTML Elements/ Responsive Design(adapt to different screen sizes and devices)

1. External Stylesheets

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

1. Internal Stylesheets:

<style>

body {

background-color: #f0f0f0;

}

</style>

1. Inline Styles:

**Typography on the web**

* + - **Typography on the web vs typography for print**

Print: Larger sizes and higher line spacing =better readability

Web: web-safe fonts, smaller font sizes and tighter line spacing to fit more content within the limited screen

* + - **Methods of adding fonts to our code**

Using: Web Fonts/ @font-face/ Directly Including Fonts/ system front

**Web inspector**

* + - **What is it? What is it used for? How can we access it?**

A developer tool in brewers

Allow user to view, inspect and debug web pages

Right-click and select "Inspect"

**CSS layout**

* + - **Block vs inline elements**

B: start on a new line and take up the full width available. <div>, <p>, <h1> - <h6>.

I: do not start on a new line and only take up as much width as necessary. <span>, <a>, <strong>, <em>.

* + - **The Box Model**

a rectangular box with content, padding, border, and margin

* + - ***box-sizing* property**

determines how the width and height of an element are calculated.

* + - **Absolute and relative units**

A: (px) (pt) (in) they are fixed and do not change with the size of the viewport.

R: (%) (em) (rem) (vw) (vh) depend on other factors, such as the parent element's size or the viewport size.

* + - ***position* property**

determines how an e. lement is positioned within its parent container

1. static: Default position
2. relative: relative to their normal position
3. absolute: elements are positioned relative to the nearest positioned ancestor.
4. fixed: relative to the viewport, do not move when the page is scrolled.
5. sticky: relative to the viewport until a specified scroll position is reached, then they behave like fixed.

**Responsive web design**

* + - **What is responsive design?**

able to view and interact across a wide range of devices and screen sizes

* + - **Viewport**

the visible area of a web page displayed within the browser window

* + - **Design/interaction changes amongst breakpoints**

where the layout and design of a webpage change to fit in different screen sizes

* + - **Media queries**

adjust styles for different screen sizes, making websites look good on all devices.

**CSS Flexbox**

* + - **What is flexbox used for? How does it work on default?**

What: a CSS layout model to layout, align, and distribute space among items in a container, even when their size is unknown or dynamic.

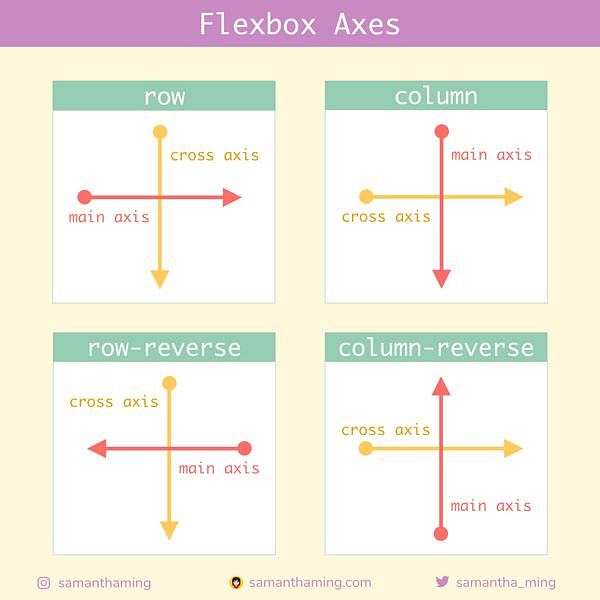
How: it lays out its items in a single row, starting from the main start edge and going towards the main end edge, and wraps items onto new lines

* + - **Flex container vs flex items**

A flex container controls flex items

* + - **Main axis vs cross axis**

main axis is the primary axis along which flex items are laid out, while the cross axis is perpendicular to the main axis.



**JavaScript intro**

* + - **What is JavaScript?**

enables interactive web elements and dynamic content on web page

* + - **Concept of holding and storing data**

JS allows data to be stored in variables, arrays, objects, etc.

* + - **Building blocks/data types of JS**

strings, numbers, booleans, arrays, objects, functions

* + - **Adding JS to a website**

<script> tags, inline or by linking to an external file

* + - **Syntax**

1. Statements and Semicolons
2. Variables and Constants

Number:

Text(string): “”

* + - **Console & console.log()**

a function that “prints out” anything that you put in the parentheses to the screen

**JS Variables**

* + - **Defining variables**

Variables are used to store data values.

“var” has function scope

“let” has block scope

“const” declares a constant whose value cannot be changed.

文本

描述已自动生成

* + - **How can variables be used?**

provide a way to give a name to a value so that it can be easily referenced and used throughout your code

1. Declaring Variables “let” “vat” “const”
2. Assigning Values =
3. Updating Variable Values
4. perform calculations or operations
5. Variable Scope, refers to where in your code a variable is accessible.
6. naming

**JS Arrays and Objects**

* + - **What is an array vs an object?**

Arrays: ordered collections of data, [ ]

Objects: unordered collections of data, keys values. { }

* + - **Using them and accessing elements within them**

A: [ ]

O: objectName.propertyName

* + - **Index number**

Starting from 0

**JS Operators and Math methods**

arithmetic, assignment, comparison, logical, etc. do math

**Conditional statements**

“if” “else if” and “else” are used to execute different code blocks based on specified conditions

**Functions**

* + - **What is a function? What is it used for?**

perform a specific task

* + - **Declaring and calling a function**

Functions are declared using the function keyword, followed by a function name and optional parameters. (the function name followed by parentheses)

**The DOM**

* + - **What is the Document Object Model?**

Document: entire web page loaded in the browser.

Object: each component of the document (e.g., elements, attributes, text).

Model: Defines how the objects are organized and accessed.

**Changing HTML & CSS**

* + - **Selecting HTML elements on JS – querySelector**

allows you to select HTML elements using CSS selectors

* + - **Changing HTML content**

“innerHTML” “textContent” “innerText”

* + - **Changing CSS styles**

modifying the style property of HTML elements

paragraph.style.color = "blue";

**Events**

* + - **Examples of event types**

Click / Mouseover / Mouseout / Keydown / Keyup / Submit / Change

* + - **addEventListener()**

attach an event handler function to an element

**Creating HTML elements using JS(from DOM)**

* + - **Creating an HTML element**

createElement(), specify the tag name of the element

* + - **Appending a created element to have it show up on the page**

appendChild(), insertBefore(), or insertAdjacentElement()

* + - **Removing a HTML element**

removeChild()

**Loops**

* + - **What are loops used for?**

1. Iterating Over Arrays
2. Repeating Tasks
3. Processing Data
4. Generating Patterns
   * + **for statement (for loop)**

initialization, condition, iteration

**文本

描述已自动生成**