**DAY-2**

**Selectors: -**

**Simple Selectors:**

1.Element Selector: The element selector selects HTML elements based on element name.

Simply write the name of the element to be selected in the element selector.

2.ID Selector: An HTML element with the ID attribute is selected with the ID selector.

To select an element with this selector, write an ‘#’ character followed by the ID-name.

3. Class Selector: This selector selects HTML elements with the specific class attribute.

To select an element with the specific class, write ‘.’ Character followed by the class name.

4.Universal Selector: This selector selects all the HTML elements on the page.

To select an element, write ‘\*’ Character.

**Combinator Selectors:**

1. Descendant Selector: This matches all elements that are Descendant of a specified elements.
2. Child Selector: This selects all elements that are the children of the specified elements.
3. Adjacent Sibling Selector: This is used to select an element that is directly after another specific element.
4. General Sibling Selector: It selects all the elements that are the next siblings of a specified element.
5. Pseudo Class Selectors: Pseudo class is used to define a special state of an element.

* Hover: This class has special effect to an element when our mouse pointer is over it.

Graphical user interface, application

Description automatically generated  
it is used to style specified part on an element.

::- it is used to add a special style to the first line of the text.

It us used to insert some content before the content on an element.

**Working with text: -**

**Font-family: -**

Fonts are specified by the property, font-family which takes different fonts separated by the ‘,’, the font names with the spaces should be enclosed in the quotes. If the browser can’t find any of the fonts in the font family list, it uses generic fonts. It a good practice to list the generic font name at the end of the font family list.

Font-style: -

A keyword that determines how the font is slanted

**Font-weight**: - A keyword or a number that determines the boldness of the font. The keywords used here are – normal, bold, bolder, lighter and numbers range is from 100-900, where 400 represent normal.

**Line-height:** A number that specify the amount of vertical space for each line. Extra spaces divided equally and above the font.

Shorthand representation of the font: -

Graphical user interface, text, application

Description automatically generated

**Text-align: -** The keyword that determine the horizontal alignment of the text.

**Text-decoration:** It determines the special decorations for the text.eg: underline, overline, line- through, null.

**CSS-box model:**

Graphical user interface

Description automatically generated

* Content-box: - It represent the content of the box.
* Padding: It is the clear area around the content. It is transparent. It is the empty space between the content and the borders.
* Border: It is used to cover the content and the padding.
* Margin: It clears the area outside he border.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

Width: it represents the width of the content area for a block element.

Height: it represents the height of the content area for a block element.

**Max height & Min Width: -**

**Max height:** This property lets you to specify an elements maximum height, means that an element can increase in width until it reaches the specific unit provided.

**Homework:** Implement box model in different boxes showing padding, margin and border and create a [tribute](https://devpractical.com/public/tribute-page-screenshot.jpg) page.

DAY-3

**Min Width**: This property defines the min. width of the element. If the content is smaller than the min. width will be applied. In other words, area will be at least of this width.

**Margins: The area between the border and the web page.**

**Margin-top** It is the space between the top border of an element and the top of the containing block.

**Margin-bottom** It is the space between the **bottom** border of an element and the **bottom** of the containing block.

**Margin-left** It is the space between the **left** border of an element and the **left** of the containing block.

**Margin-right** It is the space between the **right** border of an element and the **right** of the containing block.

**Margin in one value**: It applies the same margin to all the four sides/

**Margin two value**: The first margin applies to top and bottom and second to the left and right.

**Margin three values:** The first margin applies to the top and the second to the right and left and the third to the bottom.

**Margin four values:** The margin applies in the clockwise direction i.e. top right bottom left.

**Padding: the area between the content and the border.**

**Padding -top** It is the space between the top border of an element and the top of the content.

**Padding -bottom** It is the space between the **bottom** border of an element and the **bottom** of the content.

**Padding-left** It is the space between the **left** border of an element and the **left** of the content.

**Padding -right** It is the space between the **right** border of an element and the **right** of the content.

**Padding in one value**: It applies the same **Padding** to all the four sides/

**Padding two value**: The first **Padding** applies to top and bottom and second to the left and right.

**Padding three values:** The first **Padding** applies to the top and the second to the right and left and the third to the bottom.

**Padding four values:** The **Padding** applies in the clockwise direction i.e. top right bottom left.

**Border:** Border-width**,** Border-Style, Broder-color, **these values are applied to all sides of the borders.**

**border-[side]:** The values will Border-width, Border-Style, Broder-color, put in the exact order all these values will be applied to a specified side of the border.

**Border-width**: It takes values or keywords (Thin, medium, thick), etc.

**Border-style:** It takes keywords like dash,solid,double,etc

**Border-color:** Used to specify the color to the border.

**Background: -**

It is the short end notation for applying background color, background image,bg-attachment and background position values.

Background color: A color value or keyword that specify the color of the element background.

Background image: URL that’s points to image.

Background-repeat: -A keyword that specify if and how an image is repeated.values can be repeat,repeat-x, no-repeat.

Background-attachment: A keyword that specify wether an image scrolls with the doc or remains fixed.Values:scroll,fixed.

Background-position: It is used to specify the initial horizontal position of the image,Values:default,left,right,top,bottom.

Background-size: This property helps us to set the size of the background image.

**Flex (Flexbox):**

The flexible box layout module helps us to design flexible responsive layout structure.

Diagram

Description automatically generated

Properties:

* Display: When we assign the value flex to the display property of a container it turns into a flex container.
* Flex-direction: The main axis of the flex container is the primary axis along which the flex items are laid out placed in the direction of the main axis within the container.
* Cross-axis: the axis perpendicular to the main axis is called the cross axis. Its direction depends on the main axis direction.

**Flexbox properties for the parent or the flex container:**

* Display(display:flex): This help us to define the flex container by assigning the value When we assign the value flex to the display property of a container it turns into a flex container.
* Flex-direction: This helps in the establishing the main axis, thus defining the direction flex items are placed to the flex containers.

**Values to the flex-direction:**

1. flex-direction: row: This is the default property to lay items from left to right direction.
2. flex-direction: row-reverse: This is the default property to lay items from right to left direction.
3. flex-direction: column: This lays item from top to bottom direction.
4. flex-direction: column-reverse: This lays item from bottom to top direction.
5. flex-wrap: It allows us to wrap the items as per the need i.e. in a single line or a multiple lines.
6. Nowrap (default): It specify that all the flex items will be laid out in a single line.
7. Wrap: next items will wrap on to multiple lines.
8. Wrap-reverse: It allows next item to wrap on to multiple lines from bottom to top.
9. Flex-flow: this is a short hand form to the flex direction and the flex wrap properties.

Justify Properties:

* Justify-content: it defines how items are positioned along the main axis. It helps us to distribute extra free space. Values which can be assigned to it are:
* flex-start(default): these are positioned at the beginning of the container
* flex-end: items are positioned at the end of the container
* Flex-centre: Items are centered along the line.
* Space-between: items are evenly distributed in the lines,first items is start lines and last item is on the end line.
* space-evenly: items are distributed in such a way that spacing between any two items are equal.

Align-item property:

Baseline: Here, items are aligned in a line such that their base line align.

**Day-4**

**Align-content:** When there is extra space in the cross axis the align content aligns multiples lines in the container. It is like justify content.

**Properties: -**

* **Flex-start**
* **Flex-end**
* **Space-between**
* **Space-around**
* **Centre stretch**

**Site to practice flex-box**: <https://flexboxfroggy.com/>

Site to seek help: https://caniuse.com/

**Gap**: it helps us to control the spacing between the items.

**flex-grow**: ability for a flex item is defines the ability for a flex item to grow if necessary.

**CSS-grid:** The CSS grid layout is a 2-dimensional grid-based layout system.

A picture containing chart

Description automatically generated

* **Grid-line:**  The dividing lines that makes the structures of the grid. They can be either vertical i.e., the column grid line or the horizontal i.e., row grid line.
* **Grid-track:** It is the space between two adjacent grid lines.
* **Grid area:** The total space surrounded by the four grid lines.
* **Grid item:** It is the children of the grid container.
* **Grid cell:** It is the space between two adjacent row and two adjacent grid lines.

**Grid properties:**

1. Display: It helps us to create a grid container by setting it to the value display:grid.
2. grid-template-columns: Defines the columns and the rows of the grid with the space separated list of values.

**Text

Description automatically generated**

**Fr unit:**  it is the fraction of the free space in the grid.

**Grid gap (Gutters):**  they are used to create the gutters between columns and the rows.

* grid-row-gap
* grid-column-gap

Text

Description automatically generated

**Aligning the grid items:**

**Justify-items:** it aligns the grid items along the row axis.

Values:

1. start: It align items to the start of the row axis.
2. End: It align items to the end of the row axis.
3. Centre: It align items to the center of the row axis.
4. Stretch: It fill the whole width of the cell.

**Align-items:** It align the item along the column axis.

Values:

1. Stretch: It fill the whole height of the cell
2. Start: It align items to the start of the column axis.
3. End: It align items to the end of the column axis.
4. Centre: It align items to the center of the column axis.

**Aligning the grid items**

1. Justify-items: it aligns the grid items along the row axis. Values it takes are:

● Start/end: it aligns items to the start/end of the row axis

● Center: align items in the center of their cell

● Stretch: it fills the whole width of the cell.

2. Align-items: aligns items along the column axis.

● Stretch: it fills the whole height of the cell

● Start/End: align items to the start/end of the column axis

● Center: align items in the center of the cell

Aligning grid tracks: grid tracks can be aligned related to the grid container along the row and column axis

1. Justify content: this property aligns the grid along the row axis. Values used are:

● Start/end: aligns the grid with the starting/ending edge of the grid container.

● Center: columns are aligned at the center of the row axis.

● Space-around: the remaining space of the grid container is distributed and applied to the

start and end of each column track.

● Space-b/w

● Space-evenly

2. Align-content: this property aligns the grid along the column axis. Values used are: start, end,center,stretch,space-around,space-b/w,space-evenly.

**RWD(responsive web design)**

Media query: Media queries help us to target the browser by certain characteristics and apply styling accordingly.

**Day-5**

Javascript:

Introduction to javascript:

Include javascript to html:

1.Adding in head section:

2.Including the external file

3.inline

Syntax:

Javascript is a key sensitive language, similar to java.

**Creating identifiers:**

* Identifiers can always contain letters, numbers, underscore, dollar sign.
* Identifiers can’t start with a number.
* Identifiers are case sensitive.
* Identifiers can’t be same as reserve words.

Naming Recommendations:

* Use meaning full names
* Be consistent i.e., use camel case or underscore.
* If using underscore notation use lower case for all letters.
* If using camel case, except the first word all other words start with capital letters.

Comments: /\*comments\*/ or //

Datatypes:

1.Number: Represent an integer or decimal value that can start with a positive or negative sign.

2.String: - Represent character, string data. It begins or ends with either ‘or “.

3.Boolean: represent a Boolean value that is true or false.

4. Undefined: A value which is assigned to a variable if it is declared but not initialized.

5 Null: This value denotes nothing.

6 NaN: It stands for Not a number.

**Arithmetic expressions:**

+ plus

* minus

**Logical operators:**

* || or
* && and

**Control flow statements:**

* If statement
* If-else statement

**Functions:**

**Functions:**

**Function declaration: it is traditional way ,we start with declaring using keyword function followed by function name then the parameters.**

**Eg.**

**function myFunction(p1, p2) {**

**return p1 \* p2; // The function returns the product of p1 and p2**

**}**

**Function expression: we define a function using a variable and store the return value in that variable.**

**Function declaration:**

**Note: It is mechanism used used by js in which it moves the function declaration by function hoastness.**

**Immediately invoke company:-**