4. CSS3, CSS Flex and CSS Position

CSS3

CSS3 is a **latest** standard of CSS earlier versions (CSS2).

CSS3 - Transforms
The transform property allows you to **visually manipulate** an element by skewing, rotating, translating, or scaling.

A transformed element **doesn't affect** the **surrounding** elements, but can overlap them, just like the absolutely positioned elements. However, the **transformed** element still **takes space** in the layout at its default (un-transformed) location.

You can see list of transform property on below url:
https://www.w3schools.com/cssref/css3_pr_transform.asp

CSS Pseudo-classes
Pseudo class **selectors** are CSS selectors with a colon preceding them.

They are **immensely useful** in a variety of **situations**. Some of them are CSS3, some CSS2... it depends on each particular one.

You can see list of pseudo classes on below url:
https://www.w3schools.com/css/css_pseudo_classes.asp

CSS Transitions
CSS transitions **allows** you to **change** property values **smoothly** (from one value to another), over a given duration.

The transition effect will **start** when the specified CSS property (width) **changes** value.

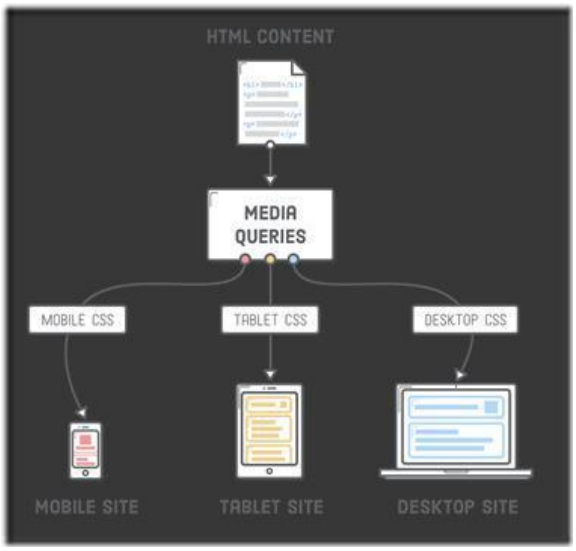
For more information checkout below url:
https://www.w3schools.com/css/css3_transitions.asp

Responsive Design

"Responsive design" refers to the idea that your website should display equally well in everything from **widescreen monitors** to **mobile phones**.

It's an **approach** to web design and development that eliminates the distinction between the **mobile-friendly version** of your website and its **desktop counterpart**. With responsive design, they're the same thing.

Responsive design is **accomplished** through CSS "**media queries**".



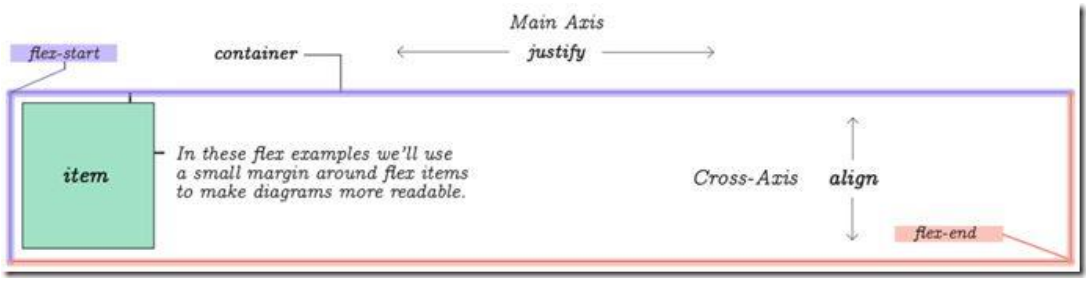
Media queries let us **present** the same HTML content as **distinct CSS layouts**. So, instead of maintaining one website for **smartphones** and an entirely unrelated site for **laptops/desktops**, we can use the **same HTML markup** (and web server) for both of them.

You can go through below URL for more information:
https://www.w3schools.com/css/css_rwd_intro.asp
https://medium.com/@mchisti/two-css-tricks-thatll-make-your-site-fully-responsive5f9efba4015e

CSS Flex

Flex is a set of rules for **automatically** stretching multiple **columns** and **rows** of content across parent container.

display: flex
Unlike many other CSS properties, in Flex you have a main container and items nested within it. Some CSS flex properties are used only on the **parent**. Others only on the items.



You can think of a flex element as a parent container with display: flex. Elements placed inside this container are called items. Each container has a **flex-start** and **flex-end** points as shown on this diagram.

While the list of items is provided in a linear way, Flex requires you to be mindful of **rows** and **columns**. For this reason, it has two coordinate axis. The horizontal axis is referred to as Main-Axis and the vertical is the Cross-Axis.

To control the behaviour of content's width and gaps between that stretch **horizontally** across the Main-Axis you will use **justify** properties. To control **vertical** behaviour of items you will use **align** properties.

If you have 3 columns and 6 items, a second row will be automatically created by Flex to accommodate for the remaining items.

If you have more than 6 items listed, even more rows will be created.

You can learn more about flex from below URL:
https://css-tricks.com/snippets/css/a-guide-to-flexbox/

CSS Position

The CSS position property defines, as the name says, how the element is positioned on the web page.

So, there are several types of positioning: static, relative, absolute, fixed, sticky, initial, and inherit. First of all, let's explain what all of these types mean.

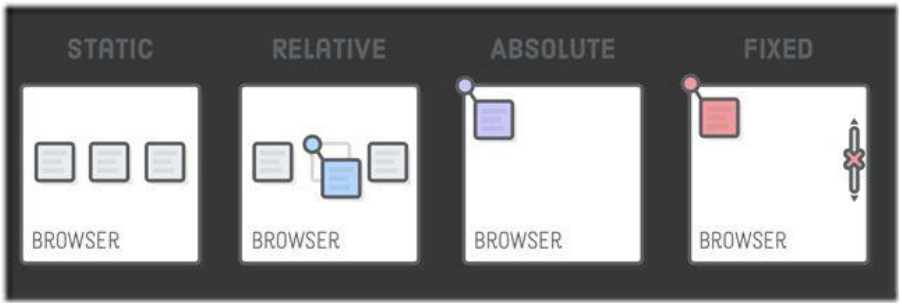
Static - this is the default value; all elements are in order as they appear in the document.

Relative - the element is positioned relative to its normal position.

Absolute - the element is positioned absolutely to its first positioned parent.

Fixed - the element is positioned related to the browser window.

Sticky - the element is positioned based on the user's scroll position.

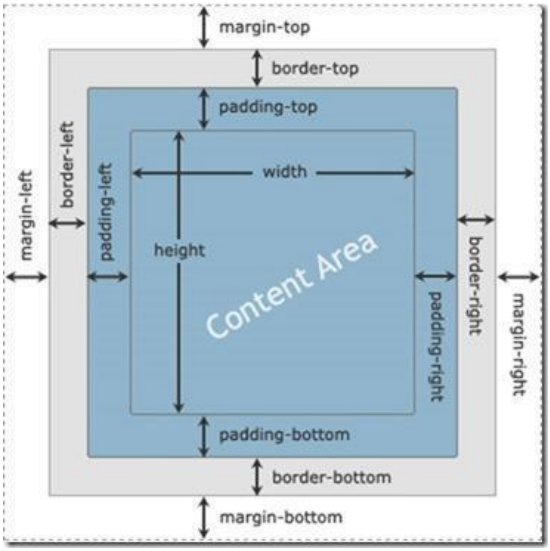


For more reference you can read below blog:
https://css-tricks.com/almanac/properties/p/position/

Box Model

Margins and Paddings dictate the spaces between elements on your website. They are very similar and have the same units as Width and Height mentioned above.

The only difference between margins and paddings is the area the exert control over. Margins affect the area outside of borders whereas paddings affect areas inside the border. It is useful to refer to the box model below:



The only difference between margins and paddings is the area the exert control over. Margins affect the area outside of borders whereas paddings affect areas inside the border. It is useful to refer to the box model below:

```
div {
    margin: 20px 10px 20px 10px;
    /* This shorthand refers to TOP, Right, Bottom, Left. It's easier to picture a clock
at 12, 3, 6 and 9 respectively */
    margin: 20px 10px 20px;
    /* This refers to Top, Left and Right, Bottom */
    margin: 20px 10px;
    /* This refers to Top and Bottom, Left and Right */
    margin: 20px;
    /* This refers to 20px worth of margin on all 4 sides */
}
```

Rules To Remember

Margins with auto on the left and right are used to center an element with a display value of block.

```
div {
    margin: 0 auto;
}
```

CSS Border

Borders have 3 different properties that you have taken care of:

border-width – width of the border. Same units as width and height.

border-style – style of the border. Usual values are solid and dashed.

border-color – color of the border. Hex, and RGB values can be used.

```
div {
    border: 1px solid black;
    /* border width, style and color */
}
```

Likewise, to margins and paddings, borders refer to all 4 sides. If you are only interested in applying borders to 1 or 2 sides, I generally prefer to stick by border-top, border-bottom, border-left or border-right.