**HTTP:**

- **HTTP** is a stateless application layer protocol used for requesting access to resources on the World Wide Web.

**Webpages:**

- Web pages are requested by web browsers, which interpret and display their contents.
- Web pages are transferred over HTTP.
- Web pages are written in **HTML (HyperText Markup Language)**.

**HTML:**

- HTML is a markup language. It describes a web page's content and structure and it is not a programming language.
- The basic structure of HTML is:

```
<!DOCTYPE HTML>
<html>
    <head>
        This is where page metadata and invisible content goes.
        Anything you want to show on the website should not go here.
    </head>

    <body>
        This is where visible page content goes.
        I.e. Anything you want to show on the website should go here.
    </body>
</html>
```

- HTML is written as a collection of **elements** which can contain content. A synonym for element is **tag**. Elements provide a structure to the document.
- Most elements are indicated by an opening tag (<>) and closing tag (</>).
  E.g. ** This is a paragraph element **
  The  is the opening tag while the  is the closing tag.
- However, some elements don't need a closing tag.
  Examples of these elements are:
  - **<br>** (Line break tag)
  - **<img … >** (Image tag)
- Elements can contain other elements.
  E.g. ** This is  important  **
  The **** tag will make whatever inside it bold. In this case, the word "important" will be bolded since it is the only text inside the **** tag.
- You can check if HTML is valid using **https://validator.w3.org/**
- The **<!DOCTYPE html>** tag tells browsers that it's an HTML file. It must be the very first thing in your HTML file.
- The **<html>** tag encloses all your HTML code, including the **<head>** and **<body>** tags. The only thing outside of the **<html>** tag is the **<!DOCTYPE html>** tag.
- The **<title>** tag will display its contents in the tab.
- The **** tag is a phrase tag. It renders as emphasized text.
  **Note:** You can use **font-style: italic;** in CSS to get the same effect as the **** tag.
- **Note:** If there are errors or missing elements/tags, the browser will not give an error. The browser will still try to load the HTML page.

- **Semantic tags** are HTML tags that indicate their expected use to both the user and the browser. Examples include:
  **<form>**
  **<h1>, <h2>, <h3>, <h4>, <h5>, <h6>**
  **<table>**
- Some advantages of semantic tags are:
  1. **Design:** Meaning of page is always the same regardless of style.
     E.g. It doesn't matter if your heading is blue or red or green, it's still an <h1> heading.
     E.g. It doesn't matter what the size of the table is, it's still a <table> tag.
  2. **Accessibility:** Screen readers can change voice tone on a tag.
     E.g. If a word is in a  tag, the screen reader would read that word louder.
  3. **Search Engine Optimization:** Density of keywords is higher when more semantic tags are used.
- **Non-semantic tags** tell nothing about its content. They are generic and have no specific purpose. Examples of non-semantic tags are  and .  is a generic inline element while  is a generic block element. They are used more for creating natural divisions throughout your page.
  **Note:** Don't visually divide anything themselves. You have to indicate how they should appear relative to other elements.

**CSS:**

- Stands for **Cascading Style Sheets**.
  Cascading means that there are priority rules for when we style a web page.
- It is a language that describes the style (layout and appearance) of web pages.
- HTML deals with the content and structure of a web page while CSS deals with the layout and appearance.
- CSS files are simply a set of rules to style different parts of a web page. In most cases, we put our CSS in a separate file and link it in our HTML file, but we can have inline CSS.
- To link a CSS file in a HTML file, put the following line inside the <head> tag:
  **<link rel="stylesheet" type="text/css" href="path/to/css/file">**
  **Note:** There is no closing tag for <link>.

- The rule is shown below:

style.css

```
selector {
    property1: value;
    property2: value;
    ...
}

..more selectors..
```

The **selector** identifies the HTML element or set of elements that we want to style.
The **property** is the layout property to assign value to.
The **value** is the value of the property.
E.g.

```
p{
   color: white;
}
```

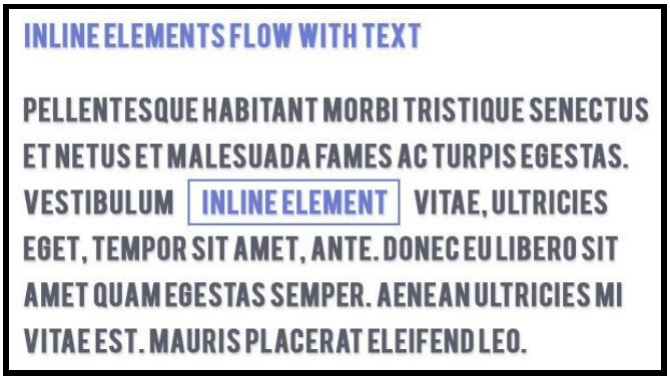
In this example, p is the selector, color is the property and white is the value of the property color. What this means is that all text inside <p> tags will have a white color.

- Some properties include:
    - Colour
    - Size
    - Shape
    - Position
    - Font
    - How to align text

**Note:** There are many more properties that we can use.

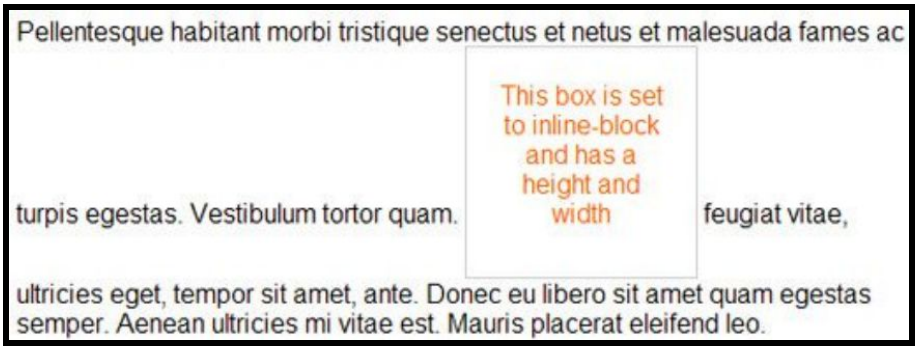
- Comments in CSS are denoted with: **/* */**. Anything placed between /* and */ will be commented out.
- Elements on a web page can be displayed in different ways.
    1. **Inline:**
        - Examples include: , <a>, <br>, 
        - It doesn't have defined width/height and can't have block elements inside it. It also doesn't create newlines.

- This is often used for changing part of a text inside a paragraph.




2. **Block:**
   - Examples include: <p>, <h1> - <h6>, <ul>
   - The height and width can be specified and changed. But by default, the width is the full width of the parent element and the height is enough to fit the content.
   - It forces creation of newlines.


3. **Inline-block:**
   - Examples include: <img>
   - These are inline elements that can have a height/width.




- **CSS/HTML classes:**
- You can define your own CSS selectors using classes.
- Classes are attributes of an HTML element.
- E.g.
  ** This is important**
  This gives this specific  tag a class attribute named "highlight".

- In CSS, you can select a specific class by putting a dot before the class name.
  E.g.
  ```
  .highlight{
      background-color: yellow;
  }
  ```
- **Note:** Classes are not unique. More than 1 HTML element can have the same class name.
- **CSS/HTML ID's:**
- The id attribute in HTML meant to be a unique identifier. Only one element should have a particular id.
  **Note:** If you have multiple elements with the same id, HTML will not generate an error or warning.
- E.g.
  ```
  <span id="highlight"> This is important</span>
  ```
- In CSS, you can select a specific id by putting a hash symbol before the id name.
  E.g.
  ```
  #highlight{
      background-color: yellow;
  }
  ```
- ID's can also be used as anchors in URLs.
  E.g. Consider http://mysite.ca/index.html#anchor. Because there is the #anchor, when you click on the link, it will auto-scroll to the position of anchor. Anchor is an id in this case.
- **Combining Selectors:**
- The **descendant selector** matches all elements that are descendants of a specified element. The first simple selector within this selector represents the parent element while the second simple selector represents the descendant element we're trying to match.
  E.g.
  ```
  p strong{
      background-color: yellow;
  }
  ```
  This applies to all  elements that are inside a <p> element.
- The **element.class selector** is used to select the specified element within the specified class.
  E.g.
  ```
  p.highlight {
      background-color: yellow;
  }
  ```
  This applies to all the <p> elements that have the class highlight.
- The multiple element selector is used to select multiple, different elements.
  E.g.
  ```
  p, strong, h1 {
      background-color: yellow;
  }
  ```
  This applies to all <p>,  and <h1> elements.

- **Conflicting Selectors:**

- When two selectors appear to conflict, the more specific selector takes precedence.
- E.g. Consider the code below:

  HTML:

```
<p>
        <em> Green or red? </em>
</p>
```

  CSS:

```
p em {
        color: green;
}

em {
        color: red;
}
```

  Since “p em” is more specific than “em”, it will be used.

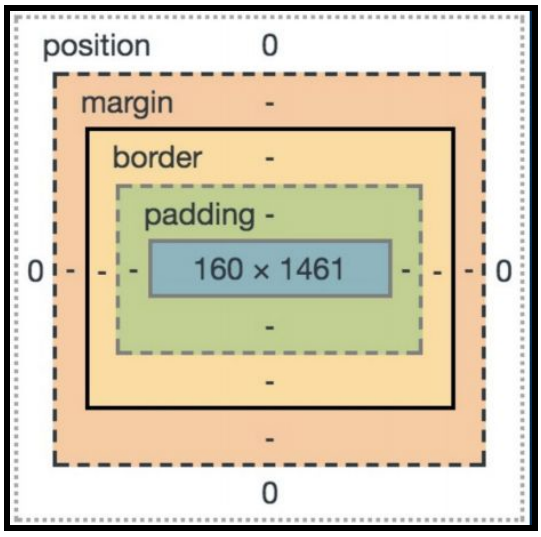
- **Specificity Precedence:**
  1. Ids
  2. Classes
  3. Elements

     **Note:** Parent elements are more specific than children elements.

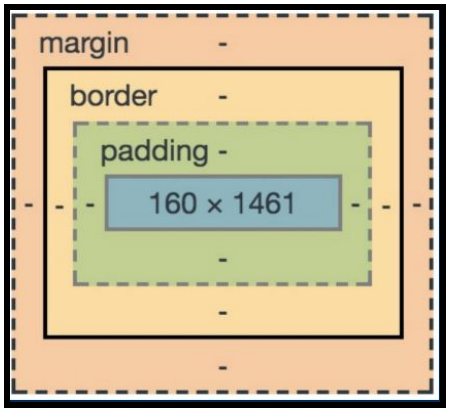
  Furthermore, if you have multiple rules with the same specificity, then the rule further down the style sheet wins.

  E.g. If you have 2 rules about the <h1> tag, the rule that’s further down wins.

- **CSS Inheritance:**
- Children elements inherit parent styles in most cases.
- If the styles of a child HTML element are not specified, the element inherits the styles of its parent.
- **Size and Position - CSS Box Model:**
- All HTML elements are considered ‘boxes’ by CSS.
- An element’s size and position are determined by values of the box model.

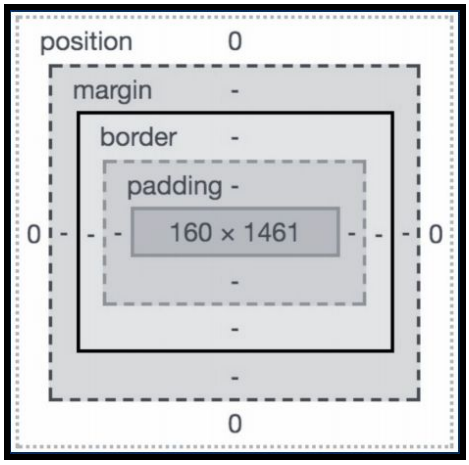


- **CSS Box Model-Size:**




- The size of an element in the box model is determined by the blue rectangle (shown in the picture above) and the 3 rectangular rings around it (padding, border and margin).
- The blue rectangle in the center is the size of the **content** of the element.
  **Note:** The numbers are in pixels.
  We can modify the size of the content using the height and width properties in CSS.
- Just outside the content is the **padding**, which is the space between the content and the border of the element. It's used to give some space between the content and the border so that the content isn't touching the border.
- The **border** comes after the padding and it can be any size you want it to be.
- Lastly, we have the **margin**, which is the area around the border (clearing space). It's used so that the elements are not touching each other.
  I.e. It's used to give some space between elements.
- All size elements are properties in CSS.
- If the content is block-displayed, we can change its size using height and width. We can specify the content size using either the number of pixels or with a percentage of the parent element it is held in.
  E.g.
  width: 20px (Using number of pixels)
  height: 50% (Using percentage of the parent element it is held in)
- For padding, we have: padding, padding-top, padding-bottom, padding-left, padding-right, etc.
- For border, we have: border, border-top, border-bottom, border-right, border-left, etc.
- For margin, we have: margin, margin-top, margin-bottom, margin-left, margin-right, etc.

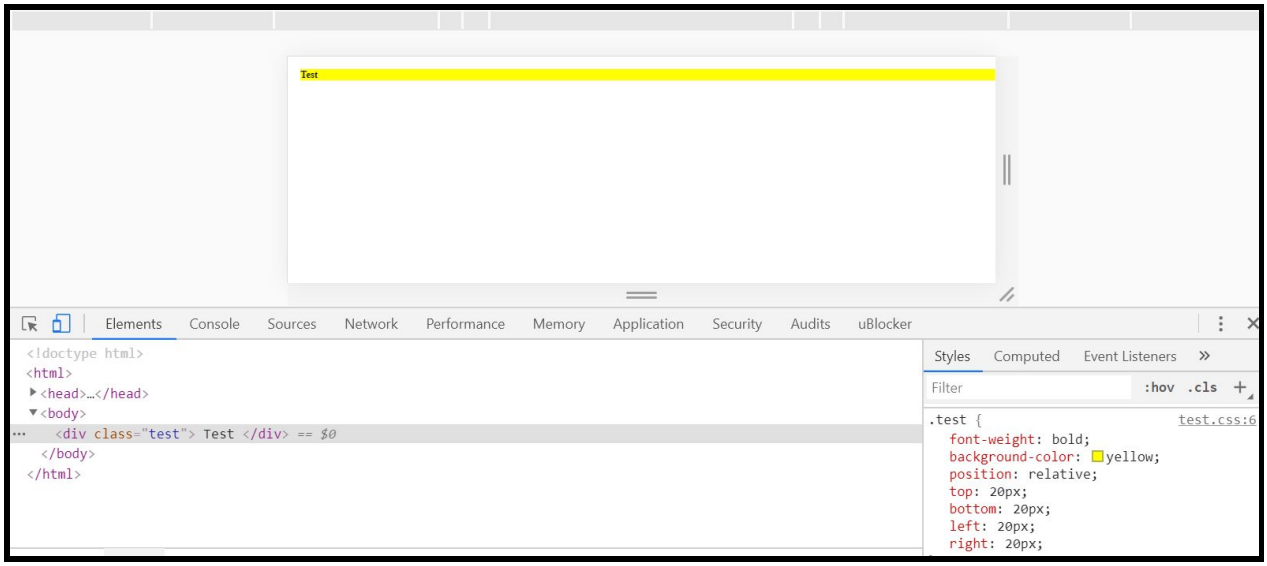
- **CSS Positioning:**



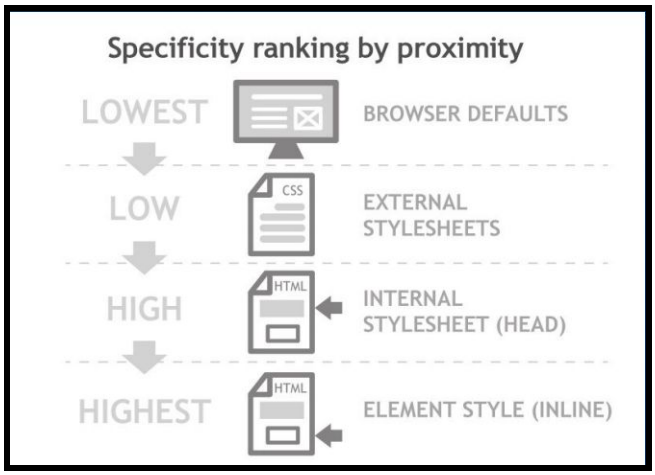

- **Positioning of elements** refers to a deviation from their **natural flow**, which is where the elements are placed by default.
- Here are a few ways you can position an element using CSS:
    1. **Static:** The default position of the element.
       I.e. Where the element is in their natural flow.
       When you put an element in HTML without changing the position type, you are by default putting it into a static position.
    2. **Fixed:** It fixes the element in one position in the viewport of the browser.
       Elements in a fixed position do not move even with scrolling.
    3. **Relative:** Allows us to change the position of an element relative to its natural location.
       E.g. We can say that we want to move an element x pixels to the right of where it is naturally located.
    4. **Absolute:** An absolutely positioned element is positioned relative to the first non-static parent. However, if an absolute positioned element has no positioned parent, it uses the document body, and moves along with page scrolling.

  **Note:** If you don't have a position, it will have, by default, the normal position.
- **Note:** Except for position: absolute, if both top and bottom are specified and are not auto, top wins. If both left and right are specified and are not auto, left wins when direction is ltr (English, horizontal Japanese, etc) and right wins when direction is rtl (Persian, Arabic, Hebrew, etc). In the case of position: absolute, if you have both top and bottom, it will span the page. Same with left and right. Top, bottom, left, and right are css properties.

E.g. The first 2 pictures below show what happens with position: absolute and the last picture shows what happens with position: relative.


- **Float:**
- The CSS float property specifies how an element should float.
- The float property is used for positioning and formatting content.
  E.g. Let an image float left to the text in a container.
- The float property can have one of the following values:
  - left: The element floats to the left of its container.
  - right: The element floats to the right of its container.
  - none: The element does not float (will be displayed just where it occurs in the text). This is default.
  - inherit: The element inherits the float value of its parent.
- **Cascade:** The order of precedence for rules on a selector.




- **Other facts:**
- **vh** stands for viewport height. 100vh means the full viewport height.
- **px** stands for pixel.