Lecture Notes

HTML & CSS

INTRODUCTION
HTML

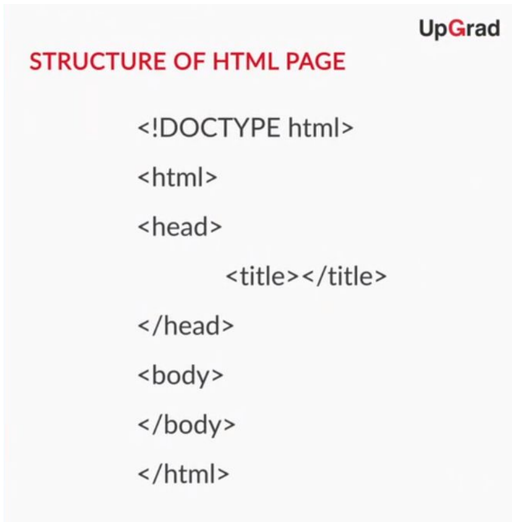
HTML stands for the **hypertext markup language**. Html is not a programming language rather it is a markup language. A markup language defines a way of formatting content in the case of HTML it tells the browser how to display web content. HTML has its own elements that make a web page in a way developers want. These HTML elements are used in the form of <tags>. Because of these tags, only HTML is called a markup language and not a programming language.

NOTE: To run an HTML code save the file with name.html extension and run it with the browser.

Technologies involved in the Front-end:

1. **HTML:** It forms the basic structure of a web page.
2. **CSS:** It deals with adding styles and making a web page look good. It deals with improving the overall appearance of a page.
3. **JS:** It adds interactive functionalities to the elements present on a web page.

Structure of HTML and tags in HTML



UpGrad

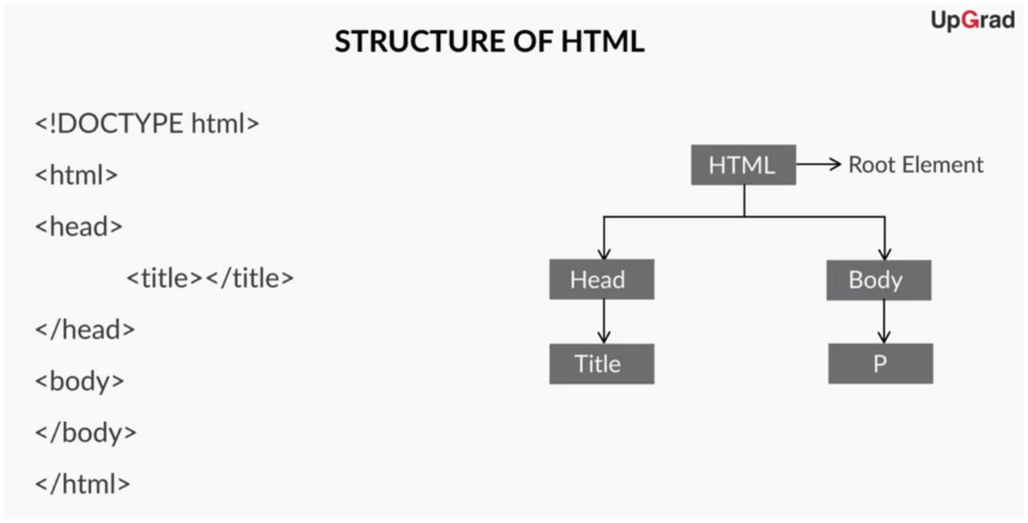
STRUCTURE OF HTML PAGE

```
<!DOCTYPE html>
<html>
<head>
        <title></title>
</head>
<body>
</body>
</html>
```

An element consists of an opening tag and a closing tag and anything between the tag is called content. This forms an HTML element:
Html element: **<opening tag> content </closing tag>**
Let's have a look at all the elements that we have used in the picture above.

1. **<!DOCTYPE html>** This version is used to tell the browser which version of HTML the page is written in. For HTML 5 we use <!DOCTYPE html>.
2. **<html>-** this element can be considered as a root element for an HTML page. A root element is the highest level parent element that consists of many child elements.

You understood this with the help of a diagram.



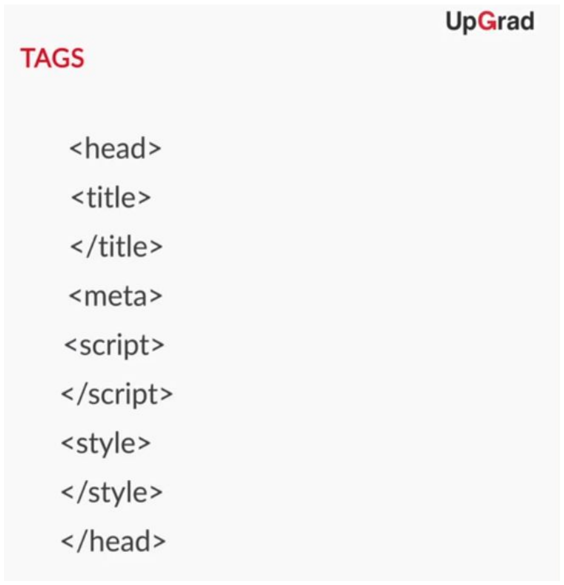
```
<!DOCTYPE html>
<html>
<head>
        <title></title>
</head>
<body>
</body>
</html>
```

<head> has child element title whereas <body> has child element <p> and all these are the child elements of <html>.
Then you learned about some HTML tags.

In a head tag, we can have these tags:-

1. **<title></title>** whatever you write in between title appears on the tab of the browser.
2. **<meta>** meta tag is a self-closing tag it does not require a closing tag. They are used by the browser to understand what the web page is all an about. It is also used to define the language of web page English, Hindi etc. They are also used to define the responsiveness of the web page.
3. **<script></script>** used to add javascript to the web page.
4. **<style></style>** used to add style elements to the web page.

These all tags come in the head tag. Have a better picture in mind from the image below


UpGrad

TAGS

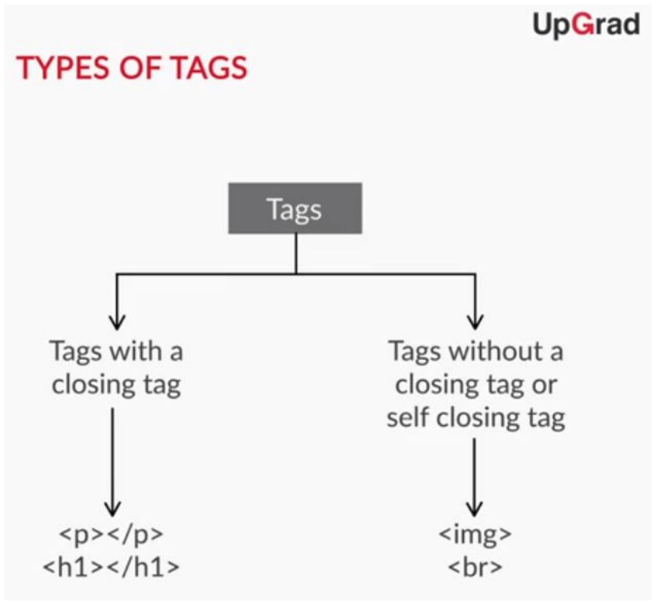
```
<head>
<title>
</title>
<meta>
<script>
</script>
<style>
</style>
</head>
```

Now you learnt about the body tag.

1. **<p></p>** this is for paragraph.
2. **h tag -** This is a heading tag. You have 6 different option from h1 to h6 with varying size. h1 is the largest and h3 being the smallest.
   a) **<h1></h1>**
   b) **<h2></h2>**
   c) **<h3></h3>**
   d) **<h4></h4>**
   e) **<h5></h5>**
   f) **<h6></h6>**
3. **<a href=””></a>** Anchor tag is used to hyperlink one page to another.
4. **<aside></aside>, <section></section>** these both are used to format the paragraphs.
5. **<img>** used to add an image to the webpage
6. **<video></video>** used to add a video to the webpage
7. **<audio></audio>** used to add audio to the webpage

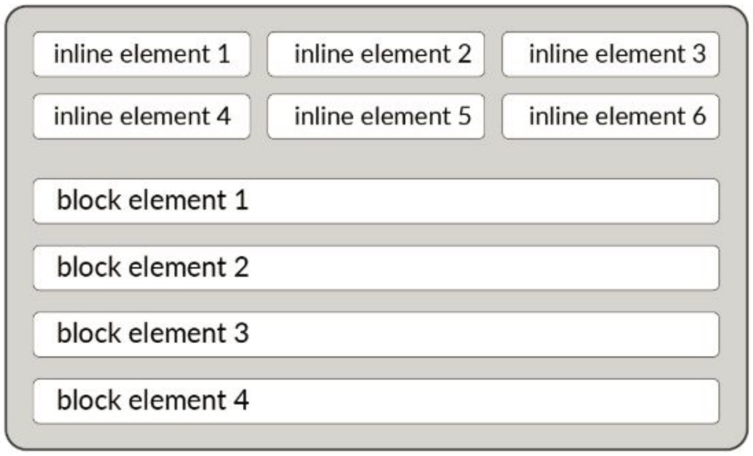
There is two type of tags :

1. Tags with a closing tag.
2. Self-closing tags.



The two types of elements:-

1. Inline- These are elements that take up only the space they require. There are no line breaks before or after them.
2. Block- These are elements that have a line break before and after them such that they take up the entire horizontal width of a row.



.

Attributes

Attributes in HTML provide additional information about a tag. An attribute is a property of a tag; a tag can have many attributes, and different tags may have different attributes.
For example, ‘<a></a>’ is called an ‘anchor’ tag. It can be used to create links on a web page. With it, you specify an **attribute** called **‘href’**, and you assign a value to it, which is the link it should go to. Here, "href" becomes the key; the link becomes the value, and both of them form an attribute together.

The syntax for this is —
<a href=""></a>

Similarly, for an **‘<img>’** tag, you provide an attribute **‘src’**, which holds a value of the location of the image in question. The tag also takes another optional attribute known as **‘alt’**, which provides a **text-based name for the image** and can be useful to vision-impaired users who use screen-reader software.

Then you learnt about some more tags

<ul><li></li></ul> tag:-

```
<ul>
    <li>list one</li>
    <li>list two</li>
    <li>list three</li>
</ul>
```

This tag is for unordered list it will present data in a form of list.In this we will see bullet points

<ol><li></li></ol>

```
<ol>
    <li>list one</li>
    <li>list two</li>
    <li>list three</li>
</ol>
```

This tag is for ordered list it will present data in a form of ordered list.In this we will see numeric indexes.

Grouping in HTML

Sometimes, you want to group individual elements in HTML together. This is done mainly when you want to apply similar CSS properties (style, colour, height etc) to a group of elements so that you don’t have to apply them individually to each one. It can also be done to give structure to your HTML code, to keep it neat.<div> and <span> tags are used for this purpose.

Difference between <div> and <span>

Div	Span
Div is a block element, which means it has a line break before and after it.	Span is an inline element, which means it only takes up the space it requires. And there are no line breaks before or after it.

Input tag

An input element can take the following values for the type ‘attribute’:

1. Button
2. Checkbox
3. Colour
4. submit
5. Tel
6. Text
7. Email
8. Date
9. Image
10. Week
11. Month
12. Number
13. Password
14. Search
15. Radio
16. Url
17. Reset

Now you made a project on the understanding oh HTML that you got from the past segments.

Introduction to Forms

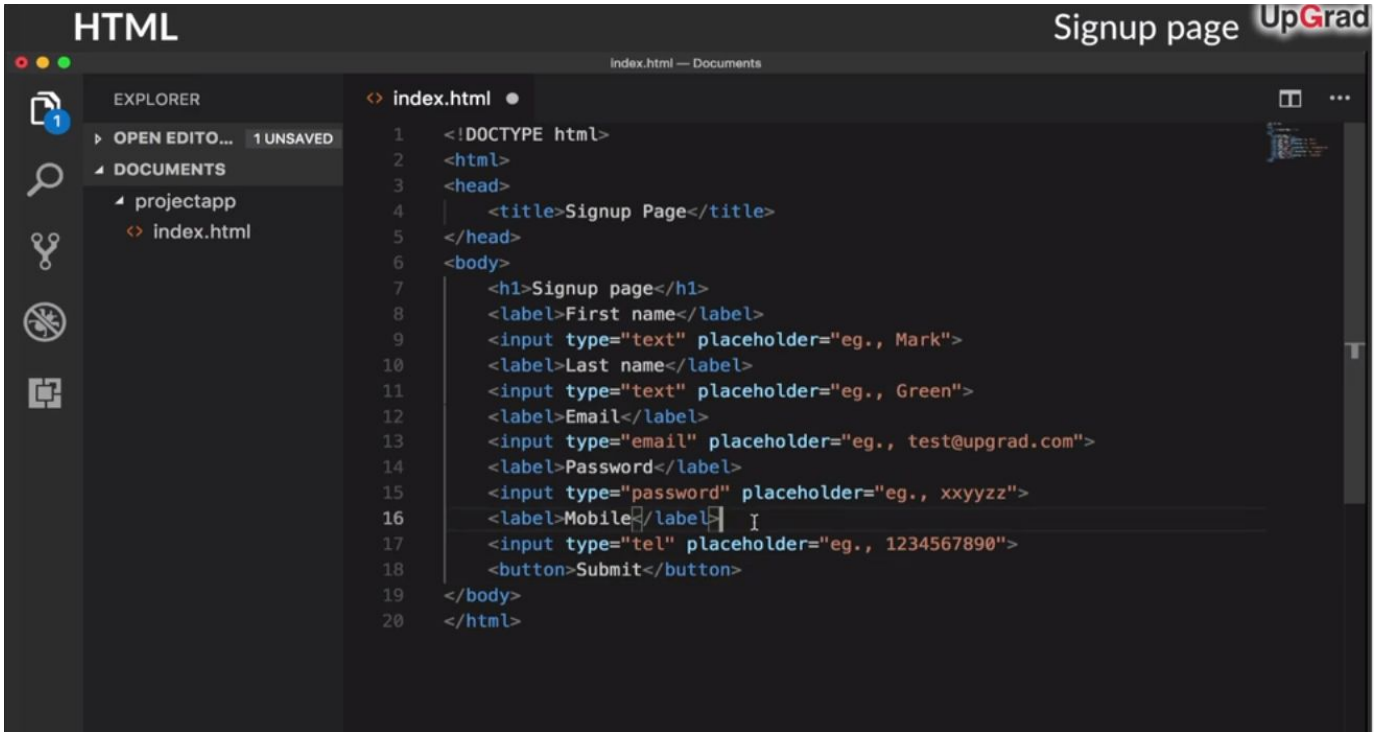
You now have all your input boxes in place, with a button for submitting the credentials entered by the user. But this will not submit the data. To integrate your sign-up page with a back-end database, you will have to wrap all the input fields in a tag called the ‘form tag’.

The <form></form> element helps in collecting data from all the input fields and sending it to the server when the user clicks ‘submit’. The ‘form tag’ uses two attributes:

1. Action: This attribute specifies the URL where the data is supposed to be sent when the user clicks on the ‘submit’ button.

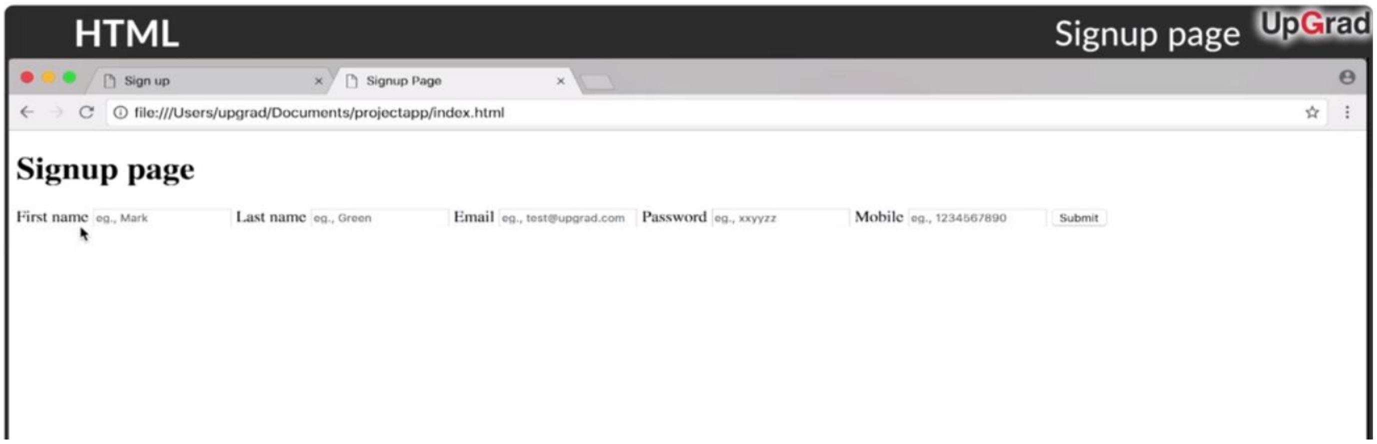
2. Method: This attribute specifies which method you want to use, such as the GET method or POST method. You shall learn more about this in one of the modules in this course itself.

eg:
<form>
Form elements such as input tags
</form>



```
<!DOCTYPE html>
<html>
<head>
    <title>Signup Page</title>
</head>
<body>
    <h1>Signup page</h1>
    <label>First name</label>
    <input type="text" placeholder="eg., Mark">
    <label>Last name</label>
    <input type="text" placeholder="eg., Green">
    <label>Email</label>
    <input type="email" placeholder="eg., test@upgrad.com">
    <label>Password</label>
    <input type="password" placeholder="eg., xxyyzz">
    <label>Mobile</label>
    <input type="tel" placeholder="eg., 1234567890">
    <button>Submit</button>
</body>
</html>
```

The output of this code will look like this



Signup page

First name eg., Mark Last name eg., Green Email eg., test@upgrad.com Password eg., xxyyzz Mobile eg., 1234567890 Submit

SUMMARY

In the session on HTML you learnt the following:

1. What is HTML
2. Why it is called “Hypertext Markup Language”
3. The basic structure of an HTML page
4. The various types of HTML tags
5. What HTML elements are
6. The difference between HTML tags and elements
7. How some tags can have a closing tag and how others do not need to have a closing tag
8. Inline and block elements in HTML and why they are called so
9. <form></form> tag and the input fields that go inside a form

Finally, with the help of the concepts above, you created a sign-up page. All of this was about HTML. Now, in the next part, you will learn about CSS and apply it to your sign-up page to make it look better.

INTRODUCTION CSS

CSS is also known as Cascading Style Sheets. It is basically a stylesheet wherein you can define how your HTML elements are going to look on-screen. You can change the position, colour, margin and many other properties of your HTML elements. You can define the CSS styling of the elements either inside the same HTML file, or you can define all of the appearance properties inside a separate .css file.

Selectors are used to identifying the element that you want to style, from all the other HTML elements, so that appropriate styling can be applied to the correct element.
They are succeeded by opening and closing curly braces. Within these braces, you can use any CSS property and give a value to that property.
Eg: **Selector {**
   color: red;
}

Now you were introduced to some of the most commonly used CSS properties.
Some of the commonly used properties are:-

1. **width**: This property specifies the width or horizontal length of an element. There are two ways in which the width is commonly defined:-
    - <length>
      You can use it to define width as an absolute value.
      E.g. width: 500px;

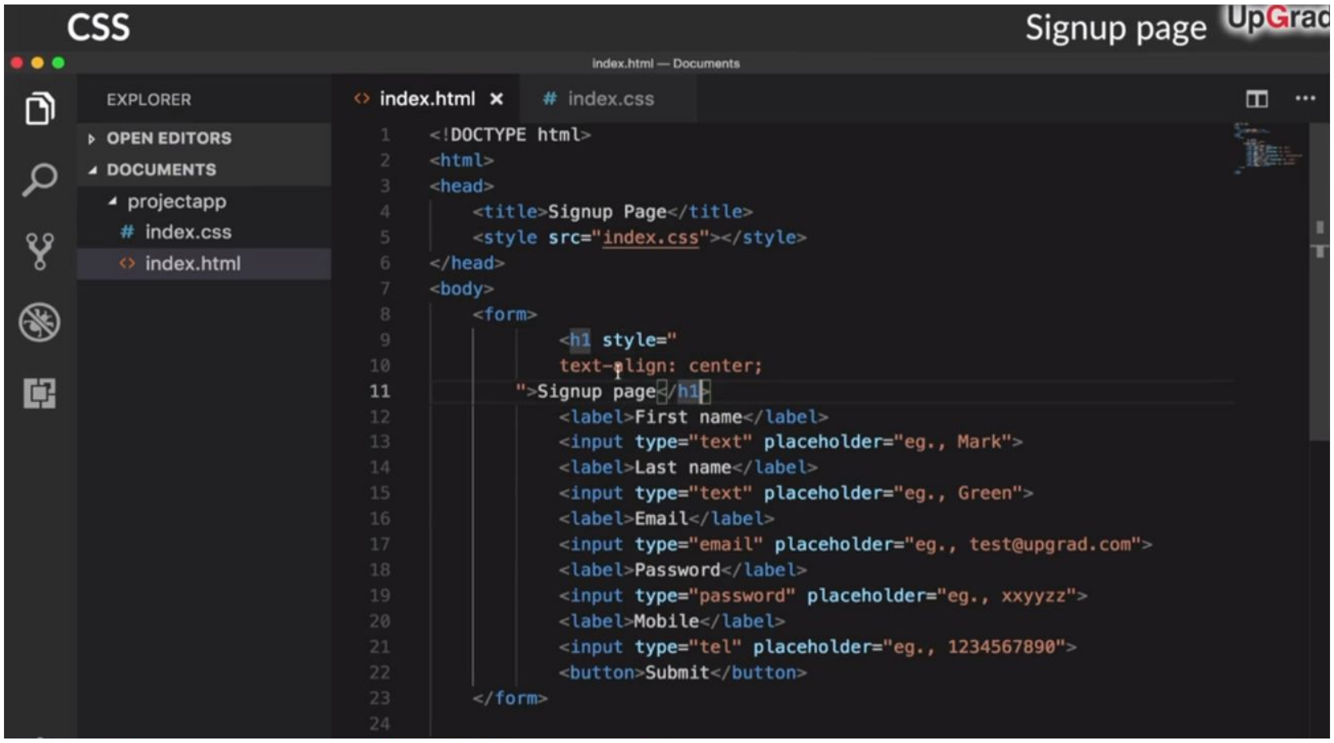
- <percentage>
  You can use it to define the width as a percentage of the containing block's width.
  E.g. width: 50%;

2. **min-width**: This property defines the minimum width an element can take.
3. **min-height**: This property defines the minimum height an element can take.
4. **max-width**: This property defines the maximum width an element can take.
5. **max-height**: This property defines the maximum height of an element.
6. **position**: This property will tell you about the type of positioning that has to be used for an element.
   - **position: static**;
     Static-positioned elements have no effect with respect to the top, bottom, left, and right property values.
   - **position: relative;**
     If you specify top, right, bottom, and left properties to a relatively-positioned element, this will cause it to be adjusted away by the specified values of these top, right, bottom, and left properties, from its normal position.
   - **position: fixed;**
     An element with 'position: fixed;' will always stay in the same place at the given coordinates, even if you scroll through the page in question.
   - **position: absolute;**
     The absolute attribute tells the browser to position the given HTML element at the exact location coordinates which are specified by the user

7. **colour**: Colour is used to give colour to text.
   - E.g. **colour: red;** or **colour: rgba(1,1,1,0.2);** or **colour: rgba(245,200,159,1);** or **colour: #ff0000;**

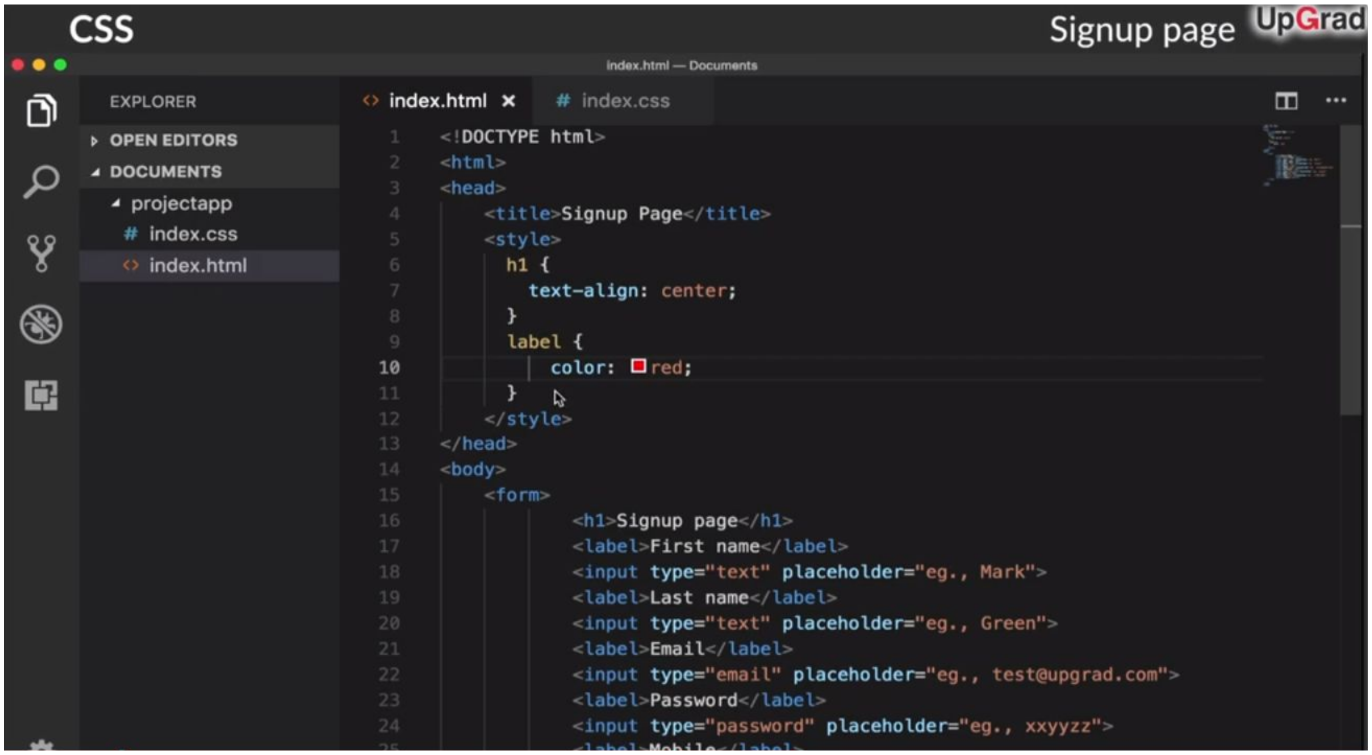
Types of selectors:
1. Id: Its use is restricted to one element. I.e in a page there can be only one id that can be unique.There cant e two different id's with the same name
2. Class: A class can be used again and again and multiple HTML tags can share the same class
3. Tag: It means we apply style directly to a tag.

Now you started to give the styling to the signup page you created in the HTML session.
First of all, you will align the main text in the centre by using the property
<style>
text-align: center;
</style>
You can use the style tag within the label or you can also place all the styles above the body.



```
<!DOCTYPE html>
<html>
<head>
    <title>Signup Page</title>
    <style src="index.css"></style>
</head>
<body>
    <form>
            <h1 style="
            text-align: center;
        ">Signup page</h1>
            <label>First name</label>
            <input type="text" placeholder="eg., Mark">
            <label>Last name</label>
            <input type="text" placeholder="eg., Green">
            <label>Email</label>
            <input type="email" placeholder="eg., test@upgrad.com">
            <label>Password</label>
            <input type="password" placeholder="eg., xxyyzz">
            <label>Mobile</label>
            <input type="tel" placeholder="eg., 1234567890">
            <button>Submit</button>
    </form>
```

In this, we placed the style tag within the label.
You can also place the style tag above the body



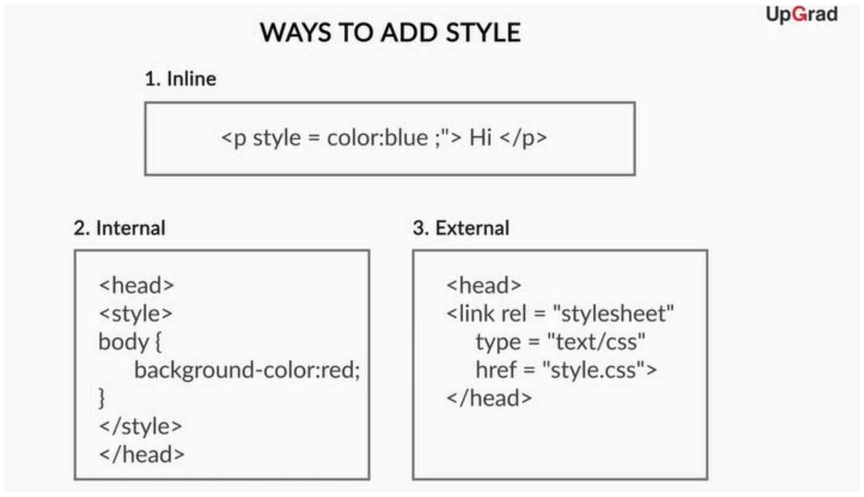
```
<!DOCTYPE html>
<html>
<head>
    <title>Signup Page</title>
    <style>
      h1 {
        text-align: center;
      }
      label {
        color: red;
      }
    </style>
</head>
<body>
    <form>
            <h1>Signup page</h1>
            <label>First name</label>
            <input type="text" placeholder="eg., Mark">
            <label>Last name</label>
            <input type="text" placeholder="eg., Green">
            <label>Email</label>
            <input type="email" placeholder="eg., test@upgrad.com">
            <label>Password</label>
            <input type="password" placeholder="eg., xxyyzz">
```

You can also give id's and classes to HTML tags and use them in the style tag to add design to the page as you had done in the signup page.
Also if you are giving a label the design and same design is also given with the help of id then the one called with id is displayed on the browser. The priority order for selection by style tag is id, class and label.

Whenever we give an id a style we call it by using a **#** hash and a class is called by a **.** (dot)
Eg:

```
<style>
#bat{
colour: red;
}
.bat{
colour: green;
}
p{
colour: blue;
}
</style>

<body>
<p id="bat" class = "bat">
My name is Batman
</body>
```

In the output, you will see My name is Batman in red colour as you know the priority of id is highest then comes the class followed by the label.



WAYS TO ADD STYLE

UpGrad

1. Inline

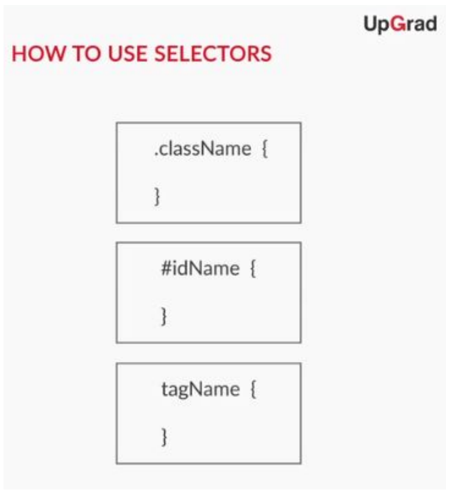
```
<p style = color:blue ;"> Hi </p>
```

2. Internal

```
<head>
<style>
body {
    background-color:red;
}
</style>
</head>
```

3. External

```
<head>
<link rel = "stylesheet"
    type = "text/css"
    href = "style.css">
</head>
```



HOW TO USE SELECTORS

UpGrad

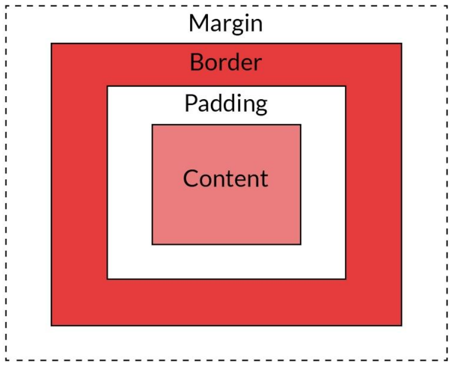
```
.className {

}
```

```
#idName {

}
```

```
tagName {

}
```

Inline has the highest priority which is followed by the internal styling and al last comes the external styling.

CSS box model



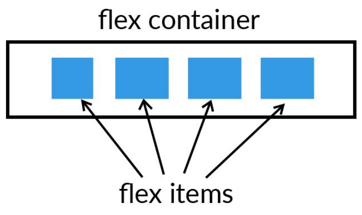
Different parts of a box model convey:

1. Content: In the content area of the box, you put content like text, images, videos, audio, etc.
2. Padding: This is the area between the border of the content and the content.
3. Border: It wraps the padding and content of the box
4. Margin: It is the blank space wrapping the border from the outside. It provides space between different HTML elements.

The ‘margin’ in the box model is the space an element can have around it whereas padding is the distance between the border of the element and the content or elements inside the border. In other words, it is internal spacing.

You can also keep your style in a separate file you will just have to add the external file in the main HTML file. Like in the signup page you have a **index.css** file in which you have kept the style tag that was there above the body in the i**ndex.htm**l file. So, in between head tag, you will have to add
<link rel=”stylesheet” href=”./index.css”> after doing this your index.css file will be linked.

Flexbox and CSS
A container, that is, an area in which content is contained, becomes flexible by setting the display property to ‘flex’. The resultant flex container is known as a ‘flexbox’, and the elements inside this flexbox are known as ‘flex’ items.

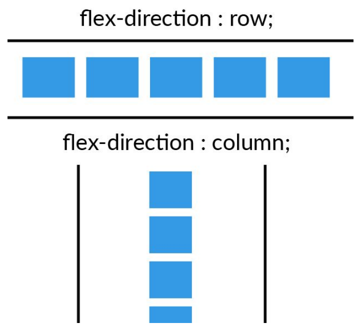


Some Flexbox properties are —

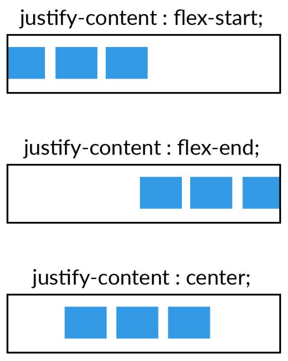
1. flex-direction
2. justify-content
3. Align-items

A) **flex-direction-**The flex-direction property tells you the direction in which the elements inside should be presented: vertically, i.e. one below the other, or horizontally, i.e. one next to the other.

1. **flex-direction: column;** This property displays elements one below the other or vertically.
2. **flex-direction: row; -** This property displays elements one next to the other or horizontally.

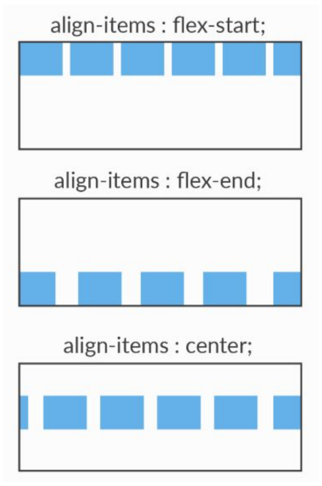


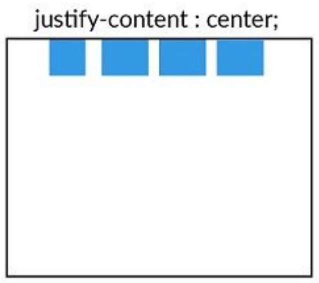
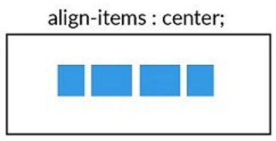
B) **justify-content** property is used to align flex items to the left, right, or centre of a container on the horizontal axis, by using the following properties and values:
1. **justify-content: flex-start;**
2. **justify-content: flex-end;**
3. **justify-content: center;**


justify-content : flex-start;
justify-content : flex-end;
justify-content : center;

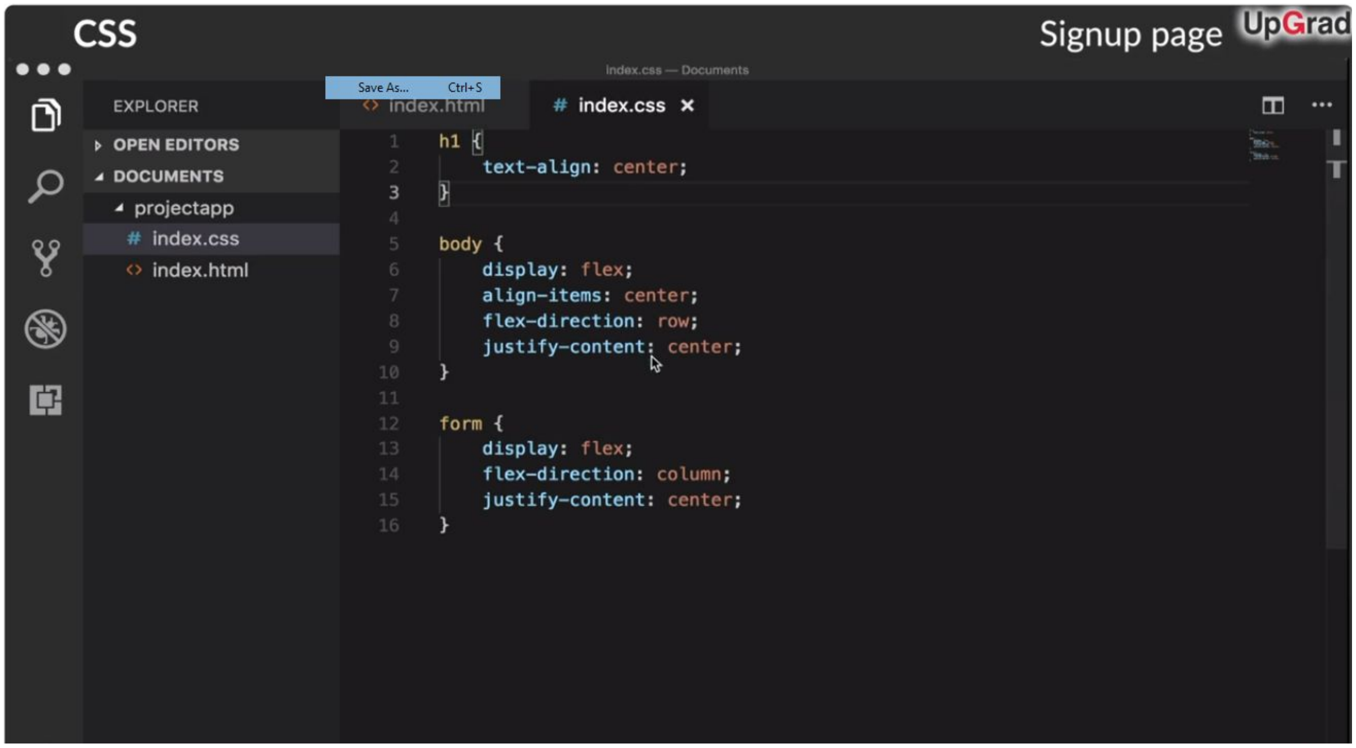
C) **Align-items-** The align-items property specifies the alignment of items inside a flexible container on the vertical axis, by using the following properties and values.

1. **align-items: flex-start;**
2. **align-items: flex-end;**
3. **align-items: center;**



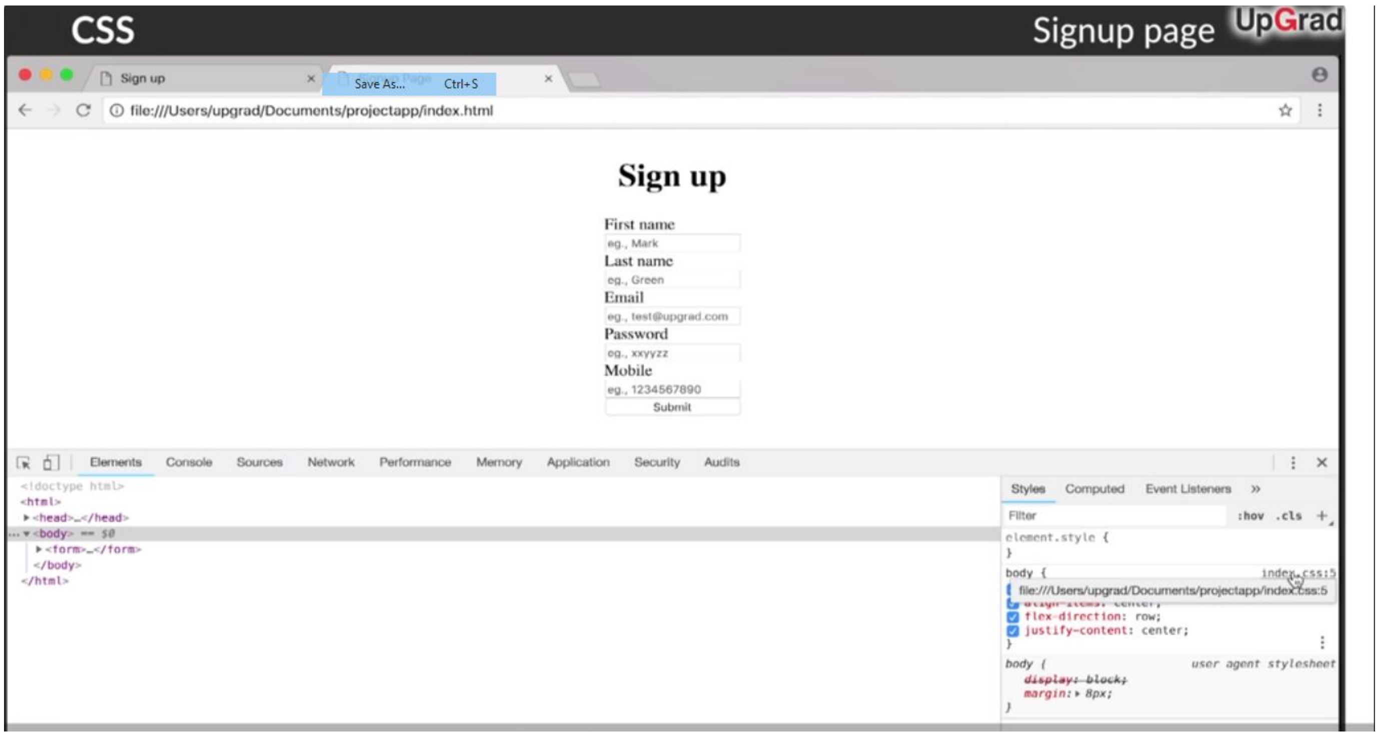
justify-content	**align-items**
Positions elements across the horizontal axis So, if you give the value ‘justify-content centre’, the content will position itself in the centre of the screen, on the horizontal axis	Positions elements across the vertical axis So, if you give the value ‘align-items centre’, the content will position itself in the centre of the screen, on the vertical axis
	

Now you used your understanding of CSS and moved all the fields on the signup page to centre.



```css
h1 {
    text-align: center;
}

body {
    display: flex;
    align-items: center;
    flex-direction: row;
    justify-content: center;
}

form {
    display: flex;
    flex-direction: column;
    justify-content: center;
}
```

This is the CSS code that you have written to move all the fields in the centre.
And the output is something like this.



Sign up

First name
eg., Mark
Last name
eg., Green
Email
eg., test@upgrad.com
Password
eg., xxyyzz
Mobile
eg., 1234567890
Submit

```
<!doctype html>
<html>
▸<head>…</head>
▾<body> == $0
  ▸<form>…</form>
  </body>
</html>
```

```
element.style {
}
body {                                   index.css:5
  file:///Users/upgrad/Documents/projectapp/index.css:5
  align-items: center;
  flex-direction: row;
  justify-content: center;
}
body {                      user agent stylesheet
  ~~display: block;~~
  margin: ▸ 8px;
}
```

After this, you applied different CSS properties and achieved the desired signup page. Some important properties used were

1. **background-colour**: This property is used to give a colour to the background of an element.
2. **width**: This property specifies the width or horizontal length of an element.

Some of the values it takes are —

width: 500px;
width: 50%;
Note: 'px' and '%' are different measurement units that determine size in CSS.

3. **font-size:** The font-size property will set the size of a font to the given value.
4. **font-weight:** The font-weight property will set the thickness or thinness of the characters in the text.
5. **font-family:** This property specifies the font for an element.
6. **cursor:** CSS has a variety of mouse cursors available. Some of them are-

cursor: pointer;
cursor: progress;
cursor: none; etc

7. **border -** This CSS property includes all the following three properties:

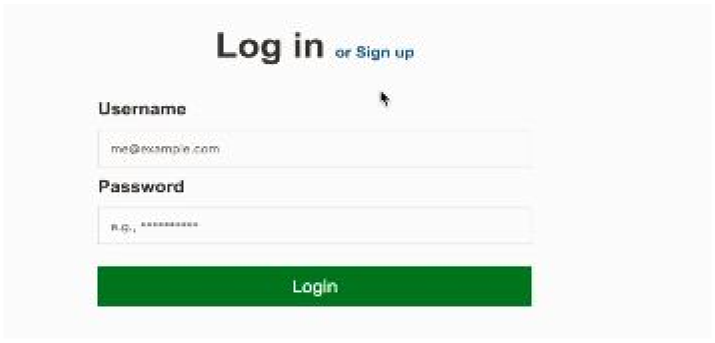
border-width
border-style
border-color

Comments in CSS

The comments in CSS can be added using /* COMMENT */.

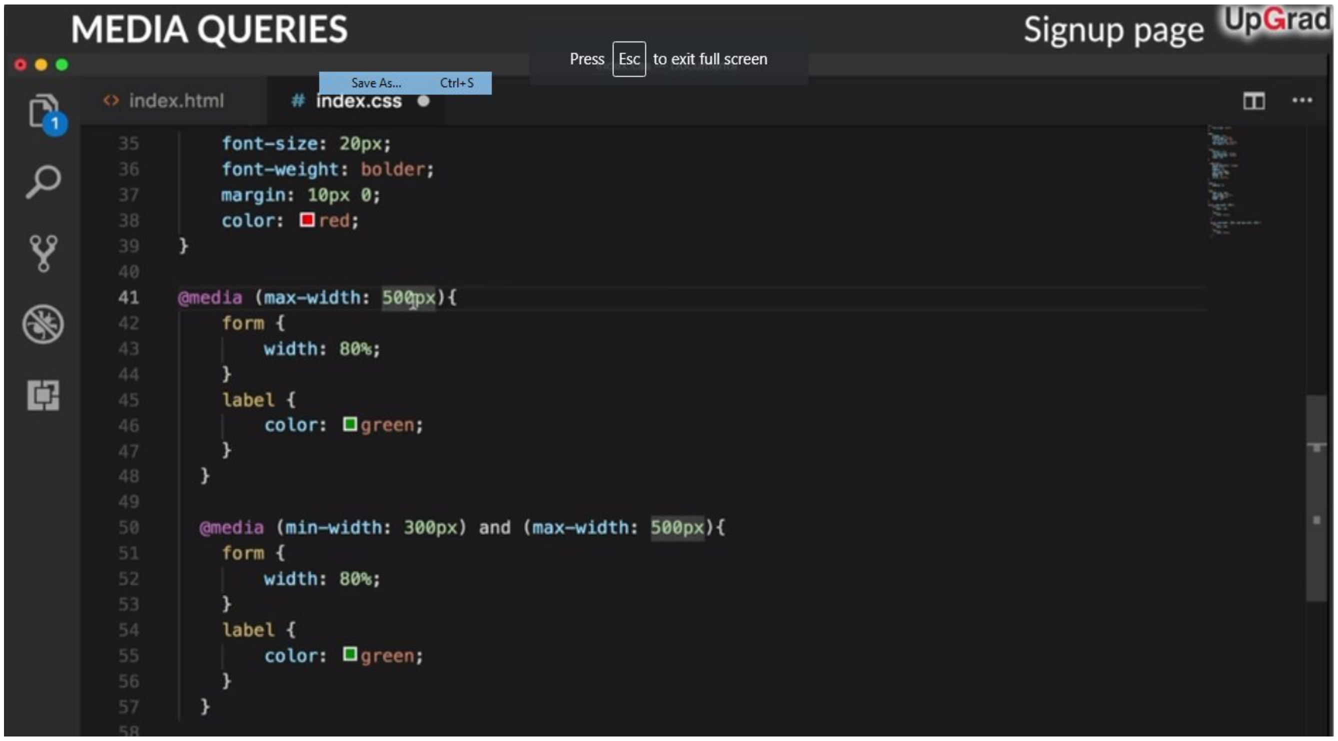
Login Page

You created the login page using the knowledge of CSS and HTML that you have gained.
The login page looked something like this.



Responsive CSS and Media Queries

Making a webpage response means that it should look nice on all screen sizes. It should not distort.
In meta tag, you have to add **<meta content=”width=device-width, initial-scale=1.0”>**
Initial-scale=1.0 tells the browser not to zoom the page and **content=”width=device-width** tells the browser that on loading the size of the HTML page must be the width of the device.
You can also add CSS according to different device sizes. See the image below.



```
    font-size: 20px;
    font-weight: bolder;
    margin: 10px 0;
    color: red;
}

@media (max-width: 500px){
    form {
        width: 80%;
    }
    label {
        color: green;
    }
}

@media (min-width: 300px) and (max-width: 500px){
    form {
        width: 80%;
    }
    label {
        color: green;
    }
}
```

SUMMARY

In this, you learnt about the following -

1. CSS or Cascading Style Sheets
2. How to apply the CSS properties to HTML elements using CSS selectors.
3. 3 types of selectors namely - class, id and tags.
4. Various CSS properties some of them being: colour, text-align, display, margin, padding etc.
5. Ways in which you can link CSS to HTML. You can do so in the following 3 ways-
    Inline CSS
    Internal CSS
    External CSS

6. With the help of the above-mentioned concepts, you gave the signup page a new look.
7. You also built a login page from scrap using HTML and then added CSS to it.
8. Finally, you learnt about the responsive CSS design and Media Queries.

This was all about CSS. There are a lot of CSS properties out there. As a developer you just need to have a basic knowledge of it, once you start working on your projects you will discover about many more CSS rules and properties.