Lesson:

Chrome dev tools for HTML

PW SKILLS

Topics Covered

- Introduction to Chrome dev tools for HTML
- Inspecting and modifying HTML elements
- Viewing and editing element attributes
- Understanding the HTML DOM structure
- Testing and debugging form elements
- Accessibility auditing and testing

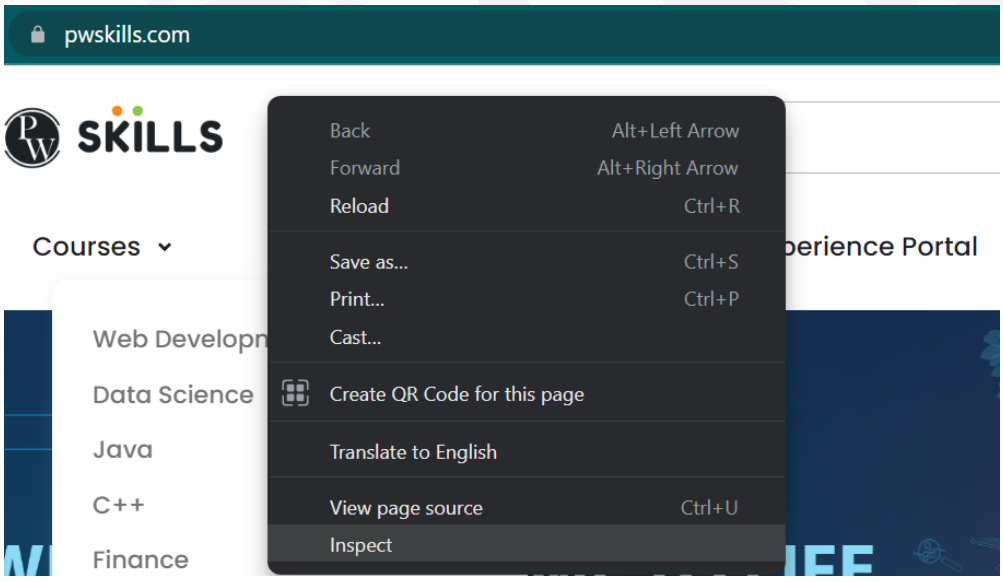
#pre-requisite - Chrome browser should be installed in the system.

Introduction to Chrome dev tools for HTML

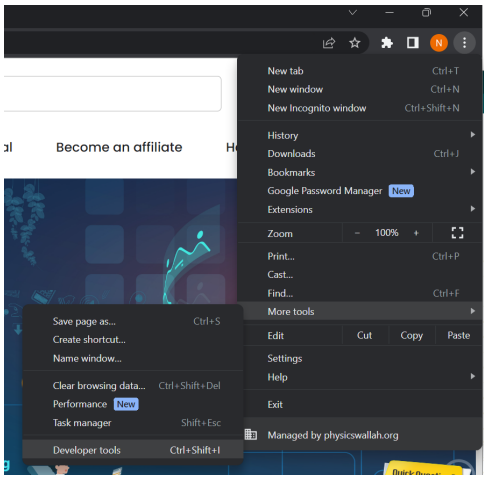
TChrome DevTools is a powerful set of web development and debugging tools built into the Google Chrome browser. It provides a range of features that help developers inspect, debug, and optimise HTML, CSS, and JavaScript code

Chrome DevTools can help you edit pages on the fly and diagnose problems quickly, which ultimately helps you build better websites, faster.
There are many ways to open Chrome DevTools, it can be accessed or opened by using the Chrome UI or keyboard shortcut.

Chrome UI i.e. open DevTools from Chrome Menus
To go to the devTools, right-click an element on a page and select **Inspect**.



Another way to open it is by clicking on the menu three dots in the top right corner of the browser. Select More tools and then select Developers tools.



Shortcut keys- open DevTools by using shortcut keys
Open the browser go desired page/website and press the shortcut key combination below to open the Dev tools

Window or Linux - Elements inspect (ctrl + shift + c)
Console inspect (ctrl + shift + j)
Your last panel(ctrl + shift + i)

Mac - Element inspect (cmd + option + c)
Console inspect (cmd + option + j)
Your last panel (cmd + option + i)

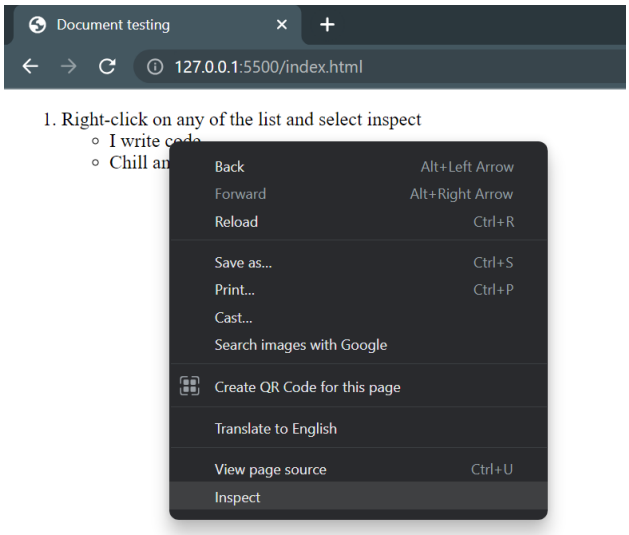
Easy way to memorise the shortcut key -

- c stands for CSS i.e the console
- j stands for JavaScript i.e the element
- i designate your choice i.e. your last panel opened.

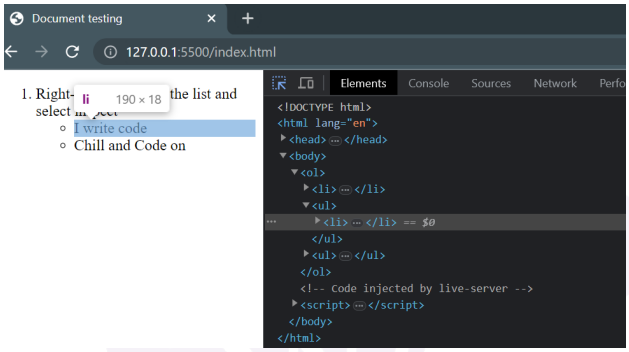
Example code

```
JavaScript
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document testing</title>
  </head>
  <body><h1>Example of Modify HTML Element</h1>
       <ol>
      <li>Right-click on any of the list and select inspect</li>
      <ul> <li>Consistency is the key to success</li> </ul>
      <ul> <li>Chill and Code on</li></ul>
       </ol>
  </body>
</html>
```

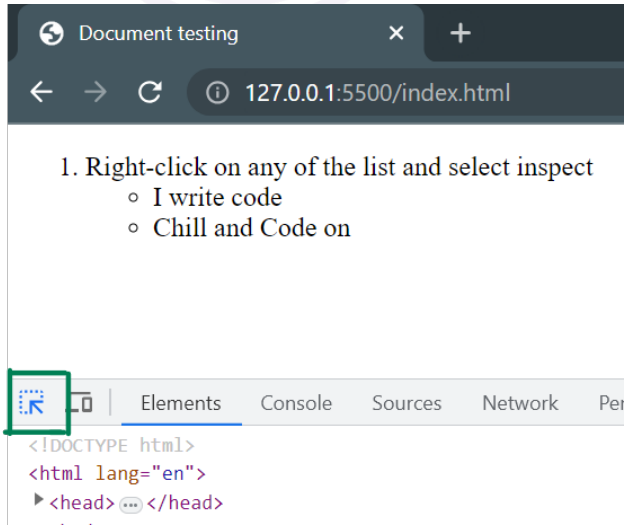
Once the live server is up and running
Right Click on **I write code** on the live server and select inspect.



Then the element panel of DevTools opens, and the **
I write code
** is highlighted in the DOM tree



Elements can be also inspected with the help of the Inspect icon in the top-left corner of DevTools



After Clicking on the inspect icons, any element on the page can be selected by hovering over them.

We can modify any element inside the Devtools element by double-clicking the HTML element.

Example Changing the HTML element from an h1 element to an h2 element

Before -

```
<body data-new-gr-c-s-check-loaded="14.1115.0" data-gr-ext-installed>
  <h1>Example of Modify HTML Element</h1> == $0
  <ol> ... </ol>
```

After -

```
<body data-new-gr-c-s-check-loaded="14.1115.0" data-gr-ext-installed>
  <h2>Example of Modify HTML Element</h2> == $0
  <ol>
```

Note - Changing of HTML element is temporary which means on a refresh of the page it will revert back to the original content.
- Just like how the HTML element can be changed the HTML element content can be also changed.

Viewing and editing element attributes

Similar to HTML elements and content that can be changed from the Chrome Devtools HTML element attributes like href, target, style, and so on, can be also viewed and edited by double-clicking on the HTML element attribute.

Example of changing HTML element attributes href value from "https://google.com" to https://pwskills.com

Before -

```
<body data-new-gr-c-s-check-loaded="14.1115.0" data-gr-ext-installed>
  <a href="https://google.com"> Link Address</a> == $0
```

After -

```
<body data-new-gr-c-s-check-loaded="14.1115.0" data-gr-ext-installed>
  <a href="https://pwskills.com"> Link Address</a> == $0
```

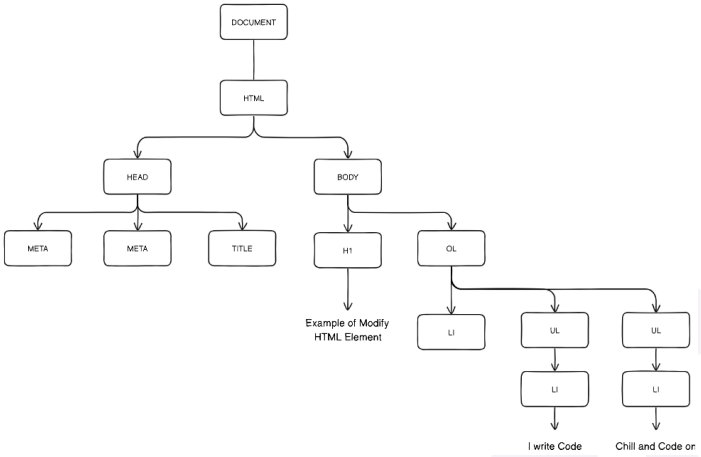
Understanding the HTML DOM structure

The HTML DOM (Document Object Model) is a programming interface for HTML and XML documents. It represents the structure of an HTML document as a tree-like structure, where each element in the document is a node, and the relationships between elements are represented by parent-child relationships. Understanding the HTML DOM structure is crucial for web developers as it allows them to access, manipulate, and interact with HTML elements using JavaScript or other scripting languages.

Here's a brief overview of the HTML DOM structure:

- Document Node - The top-level node in the DOM tree is called the "document node." which represents the entire HTML document.
- Element Nodes - represent HTML elements in the document, such as , <p>, <h1>, <img>, etc.
- Text Nodes - represent the text content within an HTML element.
- Attributes Nodes - represents the attributes of an HTML element, such as "id," "class," "src," etc.
- Parent, child, and sibling Relationships - represents each element node can have one parent node (except the document node, which has no parent).

Now, let's see the Document Object Model of the above Example code-



Tips - Chrome browser extension like an HTML Tree Generator can be used to quickly generate the DOM tree structure of the page

Testing and debugging form elements

Recap Form element,
In HTML, a form element is a container that allows users to input data and submit it to a web server for processing. It is used to create interactive sections within a webpage, enabling users to provide information or interact with the website.

Testing the form HTML element
It mainly involves manual testing of the form by interacting with each form element, verifying that the required fields are enforced and appropriate error messages are displayed when the form is submitted with missing or invalid data.

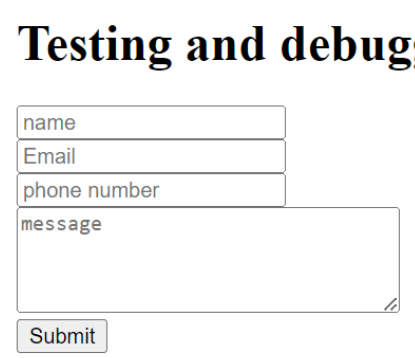
HTML5 built-in form validation attributes like required, pattern, min, max, type etc. which enables basic validation for elements like text inputs, email inputs, numbers, and more.

Example of a form HTML element with validation -

JavaScript

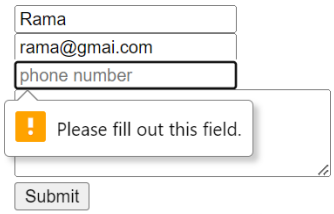
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document testing</title>
  </head>
  <body>
    <h1>Testing and debugging form element</h1>
    <form>
      <input
        type="text"
        id="name"
        name="name"
        placeholder="name"
        minlength="4"
        maxlength="26"
        required
      />
      <br />
      <input
        type="email"
        id="email"
        name="email"
        required
        placeholder="Email"
      />
      <br />
      <input
        type="tel"
        id="phone"
        name="phone"
        required
        placeholder="phone number"
        pattern="[0-9]{10}"
      />
      <br />
      <textarea
        id="message"
        name="message"
        rows="4"


        cols="30"
        required
        placeholder="message"
      ></textarea>
      <br />
      <input type="submit" value="Submit" />
    </form>
  </body>
</html>
```

Browser output –



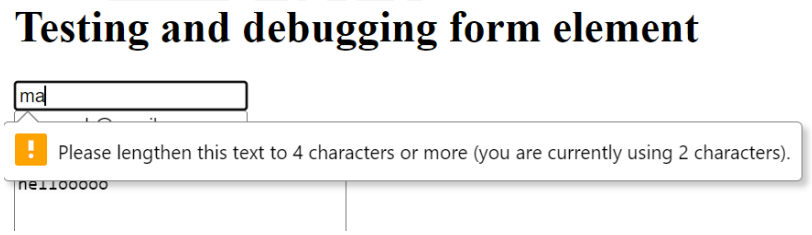
In each of the input fields inside the form element, required attribute is added so that all the fields have value. Any missing values will show a proper error message from the particular field.

For example

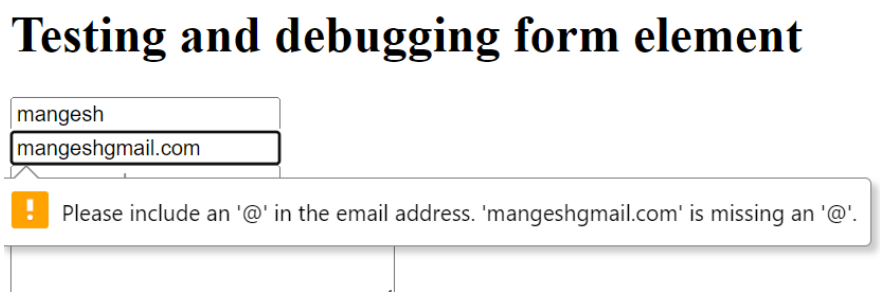


Inside the name field, the name value with less than 4 and greater than 26 characters will throw an error message. Since the input attribute is set with **minlength="4" maxlength="26"**

Example of input name field



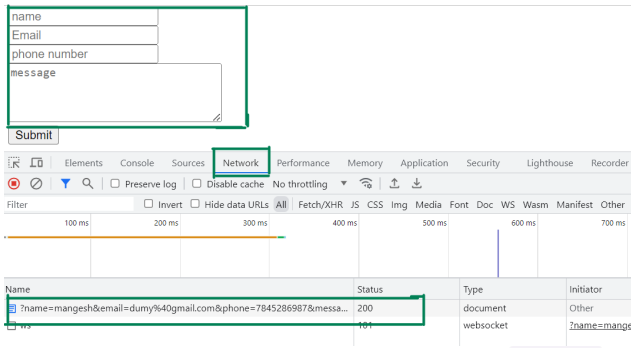
Inside the **email** field, the input values with missing **"@"** will throw an error message since it is set to **type** of email.



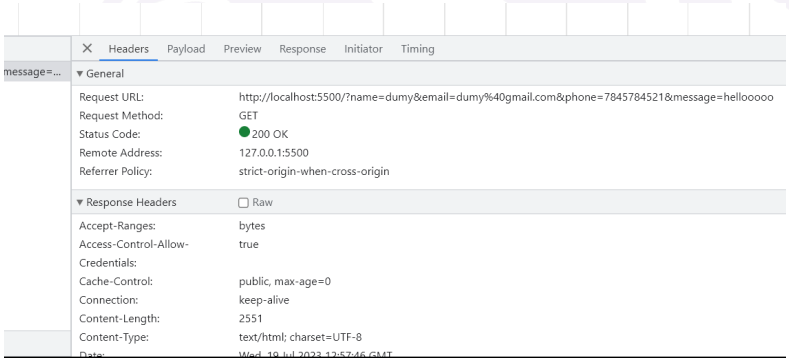
Similarly, it will throw an error message inside the phone number field when a text or string is entered rather than a number.

Debugging form element

This can be done inside the Chrome dev tools, network section. After testing and validation of the above form data, when users submit it, it is sent to the configured backend database and server and can be debugged in the network section of Chrome dev tools. After submission, all the input fields will be clear automatically.



Further clicking on the first events more details of the form data past will be displayed.



Accessibility auditing and testing

Accessibility auditing and testing are two important steps in ensuring that your website or web application is accessible to people with disabilities.

Accessibility auditing is a manual process that involves reviewing your website or web application's code and content for accessibility issues. This can be done by a developer or an accessibility expert.

Accessibility testing is an automated process that uses software to scan your website or web application for accessibility issues. This can be done by using a commercial accessibility testing tool or a free online tool

In HTML, Accessibility auditing and testing are processes that evaluate a web page's compliance with accessibility standards, identify issues that may hinder users with disabilities, and make necessary improvements to ensure that all users can access your conten

Here are some of the benefits of accessibility auditing and testing:

- Identify accessibility issues: It can help you identify accessibility issues in your website or web application before users with disabilities encounter them.
- Meet accessibility standards: It can help you to ensure that your website or web application meets accessibility standards.
- Improve usability: It can help you to improve the usability of your website or web application for everyone, not just users with disabilities.