

Image Maps in HTML with Examples

We can use HTML image maps to make clickable areas on images. An image map comprises a picture with clickable areas, where you can click on the image and it will open to a new or already specified location.

The <map> tag can include many <area> elements that define the coordinates and type of the area. You can simply link any area of a picture to other pages using the <map> tag without dividing the image.

The HTML <map> tag defines an image map. An image map is just an image with some clickable areas. The clickable areas are marked with one or more <area> tags depending on the requirement.

The idea behind an image map is that you should be able to do different activities based on where you click in the image. To make an image map, you'll need an image and some HTML code that specifies the clickable locations.

The Image

The image is inserted in Html document using the tag. The only difference from other images is that it contains a usemap attribute inside the element.

Example:

```

```

The usemap attribute value starts with a hashtag # followed by the name of the image map. The usemap attribute is used to create a connection between the image and the image map. We can use any image as an image map by using the usemap attribute inside the image element.

Create Image Map

Image map is created using a <map> element. The <map> element is used to generate an image map and is linked to the image by using the name attribute

Example:

```
<map name="deskmap">
```

The name attribute must have the identical value as the images usemap attribute then only it will work.

The Areas

After creating the map element we need to create clickable areas. A clickable area is marked using an <area> element.

```
<area shape="rect" coords="175,242,420,358" alt="Keyboard"
target="_blank" href="about.html">
```

The area can be a rectangle, circle, polygon, or a whole region; we can select any of them depending on our requirements. Within the area element, we must define the shape we are using as well as the coordinates of the area we want to make clickable.

Example:

```
<!DOCTYPE html>
<html>
<body>
<h2>Image Maps</h2>

<map name="deskmap">
<area shape="rect" coords="175,242,420,358" alt="Keyboard" target="_blank"
href="https://en.wikipedia.org/wiki/Computer_keyboard">
<area shape="rect" coords="444,251,481,357" alt="Mouse" target="_blank"
href="https://en.wikipedia.org/wiki/Computer_mouse">
<area shape="rect" coords="375,14,481,357" alt="Diary" target="_blank"
href="https://en.wikipedia.org/wiki/Book">
</map>
</body>
</html>
```

Audio in HTML with Examples:

Audio files are used to store audio data on a variety of devices, including computers, MP3 players, and cell phones. You must convert audio data into a digital format before storing it.

Encoding the raw audio data is the process of transforming audio data into a digital file. It includes extracting audio data samples and storing them in a compressed way to reduce file size.

Syntax: The basic syntax of the <audio> element

<audio> ... </audio>

Embedding Audio in HTML Document

Previously, embedding audio into a web page was difficult due to the lack of a universal standard for specifying embedded media assets like audio in web browsers. Below we'll show you how to integrate audio in your website using the latest HTML5 <audio> element.

Using the HTML5 audio Element

The HTML5 <audio> element, which was recently introduced, provides a standard way to embed audio in web pages. The audio element, on the other hand, is very new, but still, it works in most modern web browsers. The example below simply adds audio into an HTML5 document using the browser's default controls and a single source defined by the src attribute.

```
<!DOCTYPE html>
<html>
<body>
<audio controls>
<source src="birds.mp3" type="audio/mpeg">
</audio>
</body>
</html>
```

HTML <audio> Autoplay:

We can add autoplay inside the audio element to let our audio file start playing automatically.

Example:

```
<!DOCTYPE html>
<html>
<body>
<audio controls autoplay>
<source src="birds.mp3" type="audio/mpeg">
</audio>
</body>
</html>
```

Audio controls, such as play, pause, and volume, are added by the controls attribute. we can specify alternative audio files for the browser to choose from using the <source> element. The first recognized format will be used by the browser. Only browsers that do not support the <audio> element will display the text between the <audio> and </audio> tags.

Tag-Specific Attributes

The following table shows the attributes that are specific to the <audio> tag.

Attribute	Value	Description
autoplay	autoplay	This Boolean attribute indicates that the audio will begin playing automatically.
controls	controls	The browsers will display controls for controlling audio playback, such as play/pause, volume, and so on.
loop	loop	When the audio reaches the end, this Boolean attribute specifies that it will automatically start over.
muted	muted	The audio will be muted if this Boolean property is true.
src	URL	The audio file's location is specified using src

File Format	Media Type
MP3	audio/mpeg
OGG	audio/ogg
WAV	audio/wav

Videos in HTML with Examples

A video file is a collection of images displayed in a sequence to represent moving scenes. Different video codecs, such as DivX and QuickTime, are often used to encode and decode video files. A video is displayed on a web page using the HTML `<video>` element.

Syntax: The basic syntax of the `<video>` element

```
<video> ... </video>
```

Web browsers commonly support three different formats: mp4, Ogg, and WebM.

Video Controls in HTML

The browser does not display any controls for the video element by default; only the video is displayed.

```
<video src="file.mp4" >
```

This means that the audio will only play if the video is set on autoplay, and the user is unable to stop, pause, control the volume, or jump to a specific point in the video. Because the control attribute is not defined inside the video element. We can use the controls attribute inside the video element to display the built-in controls.

```
<video src="file.mp4" controls >
```

Example:

```
<!DOCTYPE html>
<html>
<style>
body{
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
background-color:mediumseagreen;
}
video{
border-radius: 8px;
border: 5px solid transparent;
}
</style>
<body>
<video width="400" controls>
<source src="/parrot.mp4" type="video/mp4">
</video>
</body>
</html>
```

It is always a good practice to include width and height attributes. If the height and width are not specified, the page may flicker while the video is loading.

Video controls, such as play, pause, and volume, are added by the controls attribute. Only browsers that don't support the <video> element will display the text between the <video> and </video> tags.

Tag-Specific Attributes

Attribute	Value	Description
load	load	Reloads the current playing video.
play	play	Start playing the video.
pause	pause	Pauses playing the current video.
autoplay	autoplay	This Boolean attribute indicates that the video will begin playing automatically.
src	URL	The video file's location is specified using src.
height	height	It is used to specify the height of the video player.
width	width	It is used to specify the width of the video player.
muted	muted	The audio will be muted if this Boolean property is true.

How to Create HTML Forms:

An HTML or a Web form helps collect user input. HTML forms can have different fields, such as text areas, text boxes, radio buttons, checkboxes, drop-down lists, file uploads, etc.

The collected data can be validated on the client browser using JavaScript. After validation of form data on the client-side, the user clicks on the Submit button in the form. After this, data is sent to the server for further processing.

Attributes for the <form> tag:

accept - This is only used when you have an <input> tag with a "file" attribute. It restricts the type of files uploaded to the server when the form is submitted. The values for this attribute should be one or more MIME types. For example: `class="example"accept="text/html, image/gif"` would only allow an HTML file or a GIF file to be uploaded.

action - This tells the form where the CGI application is located. If you are using cgiwrap, then this attribute value should be `action="http://www.engr.colostate.edu/usr-bin/cgiwrap/your_username/your_script_name"`

enctype - This sets the MIME type for the data being submitted. This is only relevant when the method attribute is "POST". The default is "application/x-www-form-urlencoded" and you will rarely need to change it. You will, however, want to set this to "multipart/form-data" if you are uploading files via your form.

method - Form are submitted with two possible methods: GET and POST. The default is GET. There are some CGI programs that require that the form be submitted via only the POST or only the GET method. So what's the difference?

GET - The data is submitted and encoded into the URL of the resulting web page. The URL takes the value of the "action" attribute, appends a "?" to it and then appends the form data set, such as `name=arthur&color=red&quest=grail` You are limited to a total of 256 bytes with this method and the result of your submission is visible in the browser's URL location bar.

POST - This method submits the data "behind the scenes" via standard input. There is no limit to the amount of bytes submitted as there is with the GET method and the data is not displayed in the browser's URL location bar.

name & id - These assign an identifier name to the form for reference within a CGI application. Newer browsers support the preferred "id" attribute, but the "name" attribute is still needed with a few older CGI programs.

target - This specifies which browser window the resultant information will be displayed. You can use a value of "_blank" to open a new browser window.

Example: `<form id="emailform" action="http://www.engr.colostate.edu/usr-bin/cgiwrap/joesmith/email.pl" method="post">`

This tag creates a form with the id of "emailform" that will submit to the program "email.php" with a "post" method.

Process to Create Forms Using HTML

HTML <form> Element

To create an HTML form, we will use the HTML <form> element. It starts with the <form> tag and ends with the </form> tag. We can add the input elements within the form tags for taking user input.

Syntax:

<form>

form elements, such as text box, textarea, etc.

</form>

Here are the different input types you can use in HTML:

- <input type="button">
- <input type="checkbox">
- <input type="color">
- <input type="date">
- <input type="datetime-local">
- <input type="email">
- <input type="file">
- <input type="hidden">
- <input type="image">
- <input type="month">
- <input type="number">
- <input type="password">
- <input type="radio">

- <input type="range">
- <input type="reset">
- <input type="search">
- <input type="submit">
- <input type="tel">
- <input type="text">
- <input type="time">
- <input type="url">
- <input type="week">

HTML <input> Element

We use the HTML <input> element to create form fields and receive input from the user. We can use various input fields to take different information from the user. Using different **Type attributes**, we can display an **<input> element** in various ways.

Example:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Text Input</title>

  </head>

<body>

  <form>

    Enter your first name <br>

    <input type="text" name="username">

  </form>

</body>

</html>
```

Common HTML Form Tags

The following are some of the commonly used HTML tags to create web forms:

Tag	Description
<form>	This tag defines an HTML form to receive input from the user.
<input>	Defines an input control.
<textarea>	Defines a multi-line input control.
<label>	Creates a caption for input elements.
<select> & <option>	Create a drop-down menu with various options. The <select> tag defines the selection or the drop-down, while the <option> tag represents all the options in the list.
<optgroup>	Creates a drop-down list of related options.
<fieldset>	Groups the related element in a web form.
<legend>	It defines a caption for the <fieldset> element.
<button>	Creates a clickable button.
<output>	It is used for defining the result of a calculation.

Password in an HTML Form

We use the type value as 'password' to take the password as input. The password entered by the user will not be visible in the password field control.

Example:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Password Input Control</title>

  </head>

  <body>

    <form>

      <p>

        <label>Username : <input type="text" /> </label>

      </p>

      <p>

        <label>Password : <input type="password" /> </label>

      </p>

    </form>

  </body>

</html>
```

Radio Button Control

Radio buttons let users select one option out of many options. We can create a Radio Button control using the HTML <input> tag and the type attribute will be radio. <input type="radio"> defines a radio button.

Example:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Radio Button Control</title>

  </head>

<body>

  <form>

    <input type = "radio" name = "Gender" value = "Male"> Male

    <input type = "radio" name = "Gender" value = "Female"> Female

  </form>

</body>

</html>
```

Submit Button Control

We can create clickable buttons in HTML using the <input>tag and setting its type attribute to Submit. Using the type="submit", we can add a submit button on the web page.

Users can submit the details provided in the form by clicking on the Submit button in an HTML form. These details are submitted to the form handler – a file on the server with a script for processing input data.

Example:

```
<!DOCTYPE html>

<html>

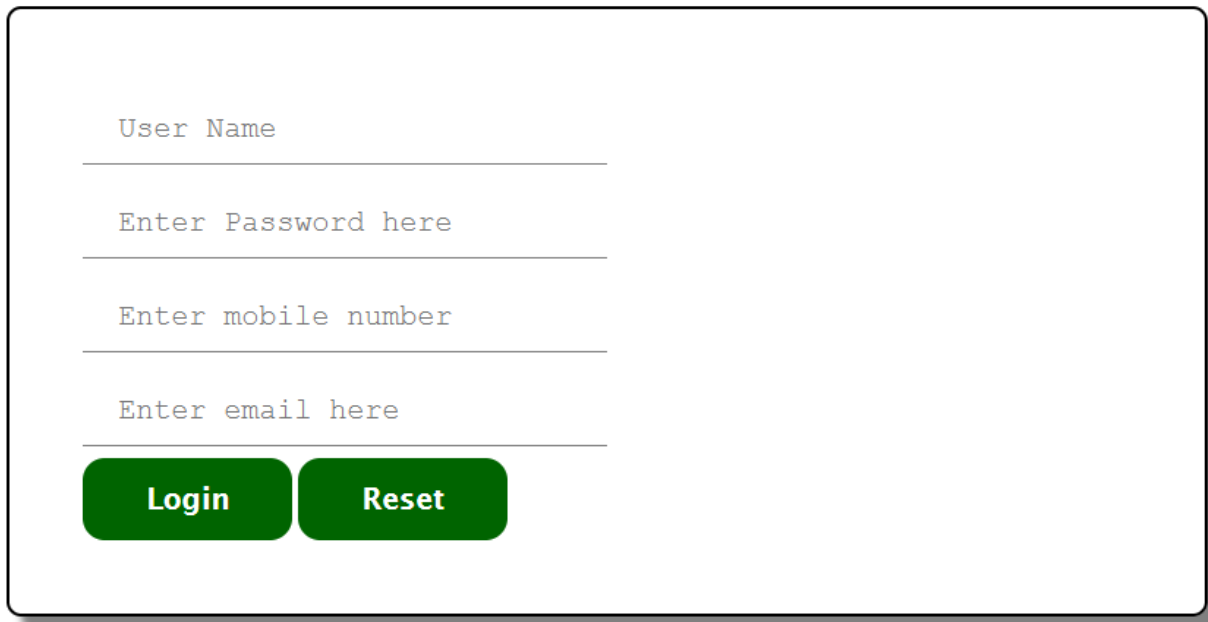
  <head>

    <title>Submit Button</title>
```

```
</head>
<body>
  <form>
    <p>
      <label>Username : <input type="text" /></label>
    </p>
    <p>
      <label>Password : <input type="password" /></label>
    </p>
    <p>
      <input type = "submit" name = "submit" value = "Submit" />
    </p>
  </form>
</body>
</html>
```

Form Design:

Create the form as bellow



The image shows a login form design within a rounded rectangular container. It features four input fields stacked vertically, each with a light gray border and a light gray placeholder text. The first field is labeled 'User Name', the second 'Enter Password here', the third 'Enter mobile number', and the fourth 'Enter email here'. Below the input fields are two green buttons with white text: 'Login' and 'Reset'.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <style>
    *
    {
      margin: 0px;
      padding: 0px;
      box-sizing: border-box;
    }
    #wrapper
    {
      width: 60%;
      margin: auto;
      border: 2px solid black;
      border-radius: 10px;
      margin-top: 40px;
      padding: 50px;
      box-shadow: 10px 10px 5px gray;
    }
  </style>
</head>
<body>
  <div id="wrapper">
    <div>
      <input type="text" value="User Name"/>
      <input type="password" value="Enter Password here"/>
      <input type="text" value="Enter mobile number"/>
      <input type="text" value="Enter email here"/>
      <button type="button" value="Login"/>
      <button type="button" value="Reset"/>
    </div>
  </div>
</body>
</html>
```

```

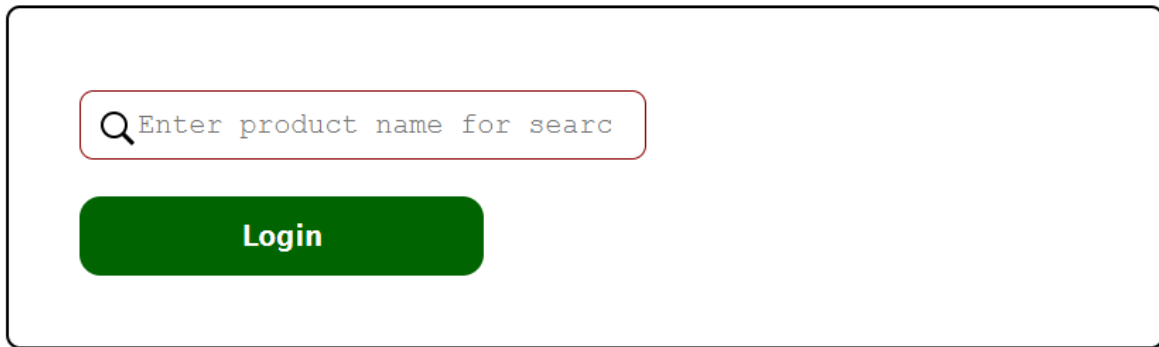
    }
    input[type=text], input[type=password],input[type=email],
    input[type=number]
    {
        width: 50%;
        padding: 12px 24px;
        margin: 8px 0;
        font-family: 'Courier New', Courier, monospace;
        font-size: 20px;
        /*border: 1px solid darkred;
        border-radius: 10px;*/
        border: none;
        border-bottom: 1px solid gray;
        /* background-color: cadetblue;*/

    }
    input[type=button], input[type=Submit], input[type=Reset]
    {
        background-color: darkgreen;
        color: white;
        width: 20%;
        padding: 16px 32px;
        font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida
Sans Unicode', Geneva, Verdana, sans-serif;
        font-size: 20px;
        font-weight: bold;
        border: none;
        border-radius: 15px;
        text-decoration: none;

    }
</style>
</head>
<body>
    <div id="wrapper">
        <form>
            <input type="text" placeholder="User Name" id="user"><br>
            <input type="password" placeholder="Enter Password here"
id="password"><br>
            <input type="number" placeholder="Enter mobile number" id="mob"><br>
            <input type="email" placeholder="Enter email here" id="email"><br>
            <input type="button" name="Login" value="Login" id="btnLogin">
            <input type="Reset" name="Reset" value="Reset" id="btnLogin">
        </form>
    </div>
</body>
</html>

```


2. Create the form as bellow.

A form with a search input and a login button. The search input is a rounded rectangle with a light gray border and a magnifying glass icon on the left. The text "Enter product name for search" is inside. Below it is a green rounded rectangle button with the text "Login" in white.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <style>
    *
    {
      margin: 0px;
      padding: 0px;
      box-sizing: border-box;
    }
    #wrapper
    {
      width: 60%;
      margin: auto;
      border: 2px solid black;
      border-radius: 10px;
      margin-top: 40px;
      padding: 50px;
      box-shadow: 10px 10px 5px gray;
    }
    input[type=text]
    {
      width: 56%;
      padding: 12px 24px;
      margin: 8px 0;
      font-family: 'Courier New', Courier, monospace;
      font-size: 20px;
      border: 1px solid darkred;
      border-radius: 10px;
```

```

        background-image: url("imgs/sr.png");
        background-position: 10px 10px;
        background-repeat: no-repeat;
        padding-left: 40px;
    }
    input[type=button], input[type=Submit], input[type=Reset]
    {
        background-color: darkgreen;
        color: white;
        width: 40%;
        padding: 16px 32px;
        font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida
Sans Unicode', Geneva, Verdana, sans-serif;
        font-size: 20px;
        font-weight: bold;
        border: none;
        border-radius: 15px;
        text-decoration: none;

    }
</style>
</head>
<body>
    <div id="wrapper">
        <form>
            <input type="text" placeholder="Enter product name for search"
id="user"><br>
            <br>
            <input type="button" name="Login" value="Login">
        </form>
    </div>
</body>
</html>

```