Experiment No. 02

Title: Design Website forms using HTML5.

Batch:A2 Roll No.:16010421063 Experiment No.: 2

Aim: To design website forms to accept data from the user through the HTML 5.0 form elements.

Resources needed:HTML 5.0 editor

Theory:

Basics of HTML Forms:

HTML forms contain **form elements**. Form elements are different types of input elements, checkboxes, radio buttons, submit buttons, and more.

For Example:
<input type="text"> defines a one-line input field for **text input.**
<input type="radio"> defines a **radio button**.

The other input elements are:

- Checkboxes
- Button
- Textarea
- Select

The different attributes of forms are:

The Action Attribute: The **action attribute** defines the action to be performed when the form is submitted.The common way to submit a form to a server, is by using a submit button.Normally, the form is submitted to a web page on a web server.
For example:
<form**action="action_page.php**">

The Method Attribute:Themethod attribute **specifies the HTTP method (**GET **or** POST**) to be used when submitting the forms:**
For example:
<formaction="action_page.php"**method="get"**> or <form action="action_page.php" **method="post"**>

A history of HTML5 forms:

The forms section of HTML5 was originally a specification titled Web Forms 2.0 that added new types of controls for forms. Started by Opera and edited by then-Opera employee Ian Hickson, it was submitted to the W3C in early 2005. The work was initially carried out under the W3C. It was then combined with the Web Applications 1.0 specification to create the basis of the breakaway Web Hypertext Application Technology Working Group (WHATWG) HTML5 specification.

Using HTML5 design principles

One of the best things about HTML5 forms is that you can use almost all of these new input types and attributes right now. They don't even need any shivs, hacks, or workarounds. That isn't to say they're all "supported" right now, but they do cool things in modern browsers that do support them-and degrade gracefully in browsers that don't understand them. This is thanks to HTML5's design principles. In this instance we're specifically referring to the principle of graceful degradation. In essence, this means that there's no excuse for not using these features right now. In fact, it means you're ahead of the curve.

HTML5 form attributes

There are 14 new attributes provided by HTML5

placeholder	autofocus
autocomplete	required
pattern	list
multiple	novalidate
formnovalidate	form
formaction	formenctype
formmethod	formtarget

1. placeholder

First up is the placeholder attribute, which allows us to set placeholder text as we would currently do in HTML4 with the value attribute. It should only be used for short descriptions. For anything longer, use the title attribute. The difference from HTML4 is that the text is only displayed when the field is empty and hasn't received focus. Once the field receives focus (e.g., you click or tab to the field), and you begin to type, the text simply disappears. It's very similar to the search box you see in Safari (see Figure 1).

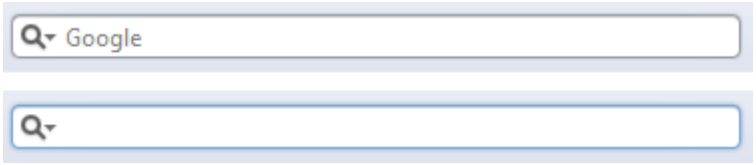


Figure 1. Browser search box in Safari without and with focus
Let's have a look at how to implement the placeholder attribute.

<input type="text"name="user-name"id="user-name"placeholder="at least 3 characters">

Figure 2 shows the placeholder attribute working in Chrome.

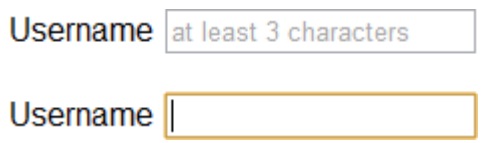


Figure 2. Placeholder attribute support in Chrome, unfocused and focused

2. **autofocus**

autofocus does exactly what it says on the tin. Adding it to an input automatically focuses that field when the page is rendered. It is a Boolean attribute (except if you are writing XHTML5; see the note) and is implemented as follows:
<input type="text"name="first-name"id="first-name" autofocus>

3. **autocomplete**

The autocomplete attribute helps users complete forms based on earlier input. The default state is set to on. This means that generally we won't have to use it. However, if you want to insist that a form field be entered each time a form is completed (as opposed to the browser autofilling the field), you would implement it like so:

<input type="text"name="tracking-code"id="tracking-code"autocomplete="off">

The autocomplete state on a field overrides any autocomplete state set on the containing form element.

4. **required**

The required attribute doesn't need much introduction; likeautofocus, it does exactly what you'd expect. By adding it to a form field, the browser requires the user to enter data into that field before submitting the form. required is a Boolean attribute, like autofocus. Let's see it in action.

<input type="text"id="given-name"name="given-name" required>

New Input Types in HTML5

- color
- date
- datetime
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

The new Elements added by HTML5

list and the datalist element

The list attribute enables the user to associate a list of options with a particular field. The value of the list attribute must be the same as the ID of a datalist element that resides in the

same document. The following example shows how list and datalist are combined (see Figure)

```
<label>Your favorite fruit:
<datalist id="fruits">
  <option value="Blackberry">Blackberry</option>
  <option value="Blackcurrant">Blackcurrant</option>
  <option value="Blueberry">Blueberry</option>
  <!-- … -->
</datalist>
If other, please specify:
  <input type="text" name="fruit" list="fruits">
</label>
```

By adding a select element inside the datalist you can provide superior graceful degradation than by simply using an option element.

```
<label>Your favorite fruit:
<datalist id="fruits">
  <select name="fruits">
    <option value="Blackberry">Blackberry</option>
    <option value="Blackcurrant">Blackcurrant</option>
    <option value="Blueberry">Blueberry</option>
    <!-- … -->
  </select>
If other, please specify:
</datalist>
  <input type="text" name="fruit" list="fruits">
</label>
```

Browser support for list and datalist is currently limited to Opera 9.5+ (see Figure 5), Chrome 20+, Internet Explorer 10 and Firefox 4+.

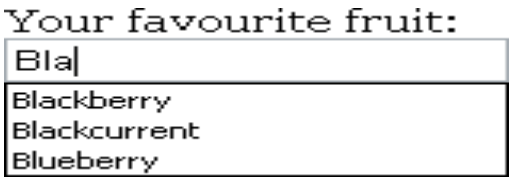


Figure 3 :Thedatalist element rendered in Opera

Attributes of the Form tag:

- Formaction
- Formenctype
- Formmethod
- Formtarget
- Novalidate
- formnovalidate

The novalidate and formnovalidate attributes indicate that the form shouldn't be validated when submitted. They are both Boolean attributes. formnovalidate can be applied to submit or image input types. The novalidate attribute can be set only on the form element.
The following example shows how to use formnovalidate:

```
<form action="process.php">
  <label for="email">Email:</label>
  <input type="text" name="email"value="gordo@example.com">
  <input type="submit" formnovalidate value="Submit">
</form>
```

And this example shows how to use novalidate:

```
<form action="process.php" novalidate>
  <label for="email">Email:</label>
  <input type="text" name="email"value="gordo@example.com">
  <input type="submit" value="Submit">
</form>
```

Activity:

Design a form (eg. Registration form/feedback form/admission form etc) with HTML 5.0 new form features.

Results: (Program printout with output / Document printout as per the format)

```
<!DOCTYPE html>
<html lang="en">
   <head>
       <meta charset="UTF-8" />
       <meta http-equiv="X-UA-Compatible" content="IE=edge" />
       <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
       <title>Document</title>
       <!-- <link href="./style.css" rel="stylesheet" /> -->
   </head>
   <body>
       <form onsubmit="myFunction(event)" action="#" class="form">
           <div style="display: flex; flex-direction: column; width: 60%;
gap: 2vh">
               <label for="name"> Enter Username </label>

               <input
                   type="text"
                   name="username"
                   id="username"
                   placeholder="enter username"
                   required
                   autofocus
               />

               <label for="name"> Enter Password </label>
               <input type="password" name="password" id="pass" required />

               <label for="color">Choose colour </label>
```

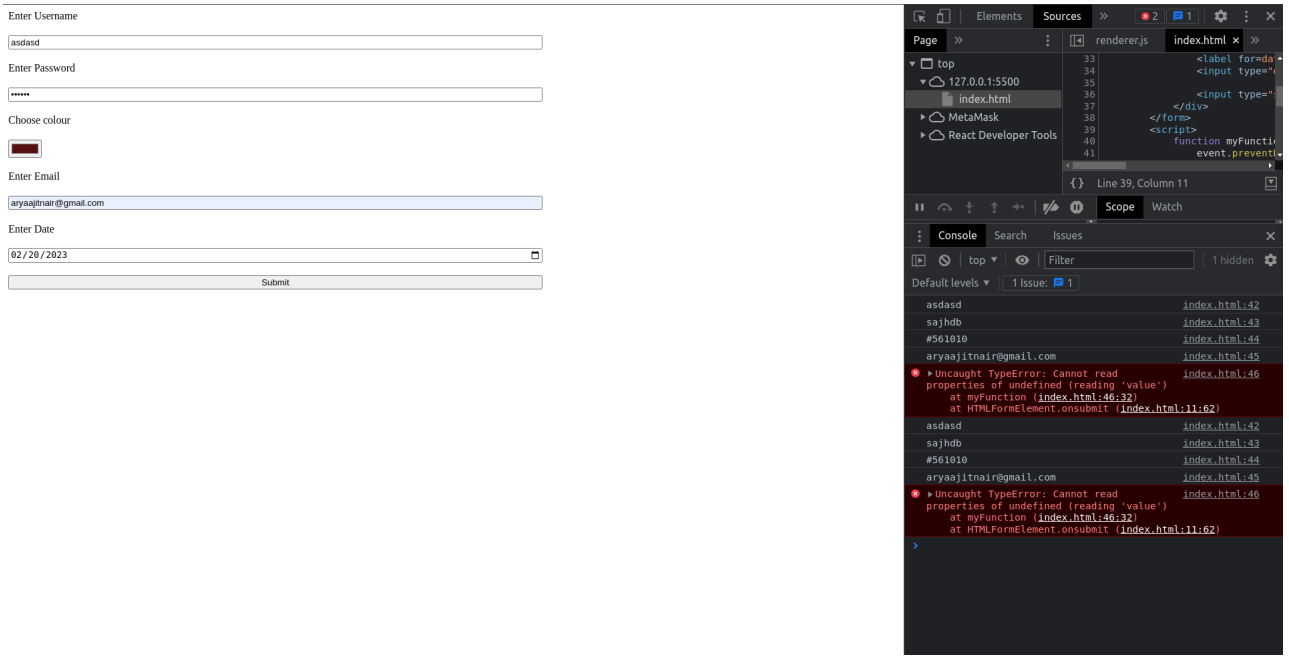
```
            <input type="color" name="color" id="color" required />

            <label for="email">Enter Email </label>
            <input type="email" name="email" id="email" required />

            <label for=datel">Enter Date </label>
            <input type="date" name="date" id="date" required />

            <input type="submit" value="Submit" />
        </div>
    </form>
    <script>
        function myFunction(event) {
            event.preventDefault();
            console.log(event.target["username"].value);
            console.log(event.target["pass"].value);
            console.log(event.target["color"].value);
            console.log(event.target["email"].value);
            console.log(event.target["date"].value);

        }
    </script>
</body>
</html>
```



Questions:

1. **What is the use of multiple in list and datalist element?**

The multiple attribute is used in HTML's <select> element when you want to allow the user to select multiple options from a list. When multiple is used, the user can select more than one option by holding down the Ctrl key (Windows) or the Command key (Mac) while clicking the desired options.

2. **What is the importance of pattern attribute?**
   The pattern attribute is an HTML attribute that is used to specify a regular expression pattern that an input field's value must match. This attribute is used primarily with the input element's type attribute set to text, search, tel, url, or email. The pattern attribute is important because it can help ensure that the user enters data in the correct format, such as when entering a phone number or an email address. By providing a regular expression pattern that an input field's value must match, the web developer can enforce a certain format for the user input.

3. **What are the three types of button that can be used in form?**
   In HTML forms, there are three types of buttons that can be used:

   Submit button: The submit button is used to submit the form data to the server for processing. When the user clicks on a submit button, the form data is sent to the server for processing.
   Example code:
   <input type="submit" value="Submit">

   Reset button: The reset button is used to reset the form fields to their original values. When the user clicks on a reset button, all the form fields are reset to their initial values.
   Example code:
   <input type="reset" value="Reset">

   Button: The button element is used to create a generic button that can be used for a variety of purposes, such as triggering a JavaScript function or opening a link. Unlike the submit and reset buttons, the button element does not perform any default action.
   Example code:
   <button type="button">Click Me</button>

   It's worth noting that buttons can also be styled using CSS to give them a specific appearance and behavior, and JavaScript can be used to add functionality to buttons

Outcomes:

CO 2 Create Web pages using HTML 5 and CSS.

Conclusion:
(Conclusion to be based on the outcomes

achieved)

Understood and implemented the forms also logged

the data into console

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

- "HTML5: Black Book", Dreamtech Publication.
- "Web Technologies: Black Book", Dreamtech Publication.
- http://www.w3schools.com