Web Technologies and Programming

Lecture 20

History, Navigator, Screen and Form Objects

Summary of Previous Lecture

- **Controlling the background dynamically**
  - **Bgcolor**
    - **SET For a document**
    - **SET For a specific **
    - **SET the Text Color**
    - **Return background color of a specific element**
    - **Return background color of a document:**
  - **Background**
    - **background color for a document**
    - **background image for a document**
    - **Set a background-image to no-repeat**
    - **Set the background-image to be fixed**
    - Change the position of a background-image

Summary of Previous Lecture

- **Working with images**
  - **Access an Image Object**
  - **On MouseOut**
  - **On MouseOver**
  - **Image Rollover**
  - **Image Preview**
  - **Image Slide Show**

- **Date object**
  - **Digital Clock**

Today's Lecture Outline

- **History object**
- **Navigator object**
- **Screen object**
- **Form object**

1. The History object

- The history object contains the URLs visited by the user (within a browser window)
- The history object is part of the window object and is accessed through the window.history property
- Used to move forward and backward through the visitor's browsing history
- All major browsers support it.

1. The History object…

- **History object properties:**
  - **Length: Returns the number of URLs in the history list**
- **History object methods:**
  - **back(): Loads the previous URL in the history list**
  - **forward(): Loads the next URL in the history list**
  - **go(): Loads a specific URL from the history list**
- **The window.history object can be written without the window prefix.**

1. The History object…

```
<html>
<head>
    <title> History Object </title>
    <script language="javascript">
        document.write(history.length)
        function goBack()
        {
            window.history.back()
        }

        function goForward()
        {
            window.history.forward()
        }
    </script>
</head>
```

Writes History Length on Webpage

Go Back Function

Go Forward Function

1. The History object…

```
<body>
    <h1> This Is The First Page </h1>

    <a href="history1.html"> Go To Next Page </a>
    <br>
    <input type="button" value="Go Back!" onclick="goBack()">
    <input type="button" value="Go Forward!" onclick="goForward()">
</body>
</html>
```

Body Contents

Call to Go Back Function

Call to Go Forward Function

1. The History object…

2. The Navigator object

- The navigator object contains information about the visitor's browser

- It also provides several properties that assist in the detection of various elements of the visitor’s browser and environment

2. The Navigator object…

- **Navigator object properties:**
  - **appCodeName: Returns the code name of the browser**
  - **appName: Returns the name of the browser**
  - **appVersion: Returns the version information of the browser**
    - **The properties appName and appCodeName return the name of the browse**
- **Navigator object methods:**
  - **javaEnabled(): Specifies whether or not the browser has Java enabled**

2.1 Detecting Users browser

- **Used to write browser specific code**
- **Can also be used to restrict users to use a specific browser**

2.1 Detecting Users browser…

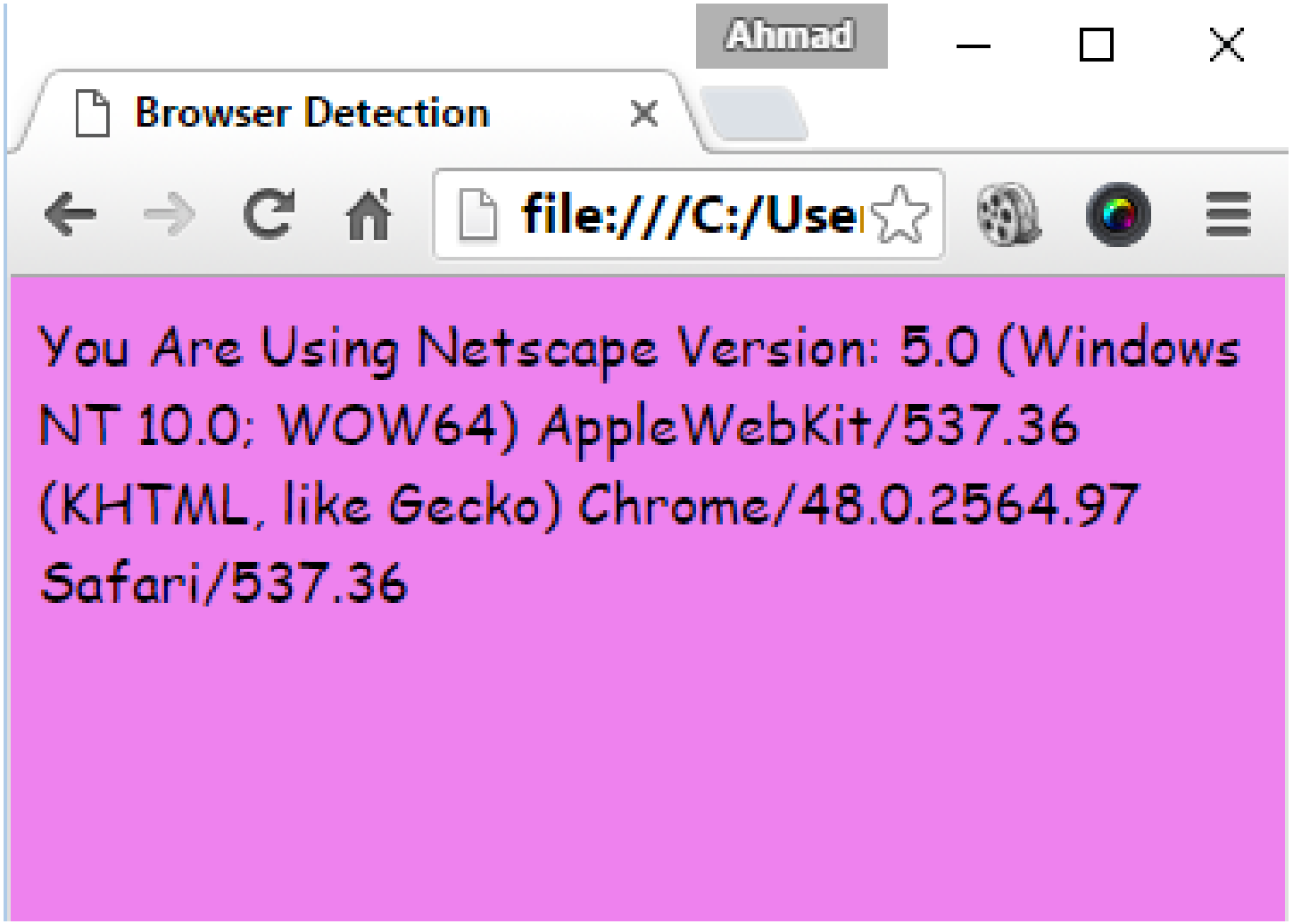
```
<html>
<head>
    <title> Browser Detection </title>
</head>

<body bgcolor="Violet">
    <script language="javascript">
        var browsername = window.navigator.appName
        var browserversion = window.navigator.appVersion

        document.write("You Are Using " + browsername + " Version: " + browserversion)
    </script>
</body>
</html>
```

Finding Browser Name

Finding Browser Version

Writing Browser Information

2.1 Detecting Users browser…



2.2 The Browser Engine…

The property product returns the engine name of the browser:

Example

```
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML =
navigator.product;
</script>
```

2.3 The Browser Platform

The property **platform** returns the browser platform (operating system):

Example

```
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
navigator.platform;
</script>
```

2.3 The Browser Language

The property **language** returns the browser's language:

Example

```
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML =
navigator.language;
</script>
```

3. The Screen object

- **The screen object contains information about the visitor's screen**

- **You might need this information to determine which images to display or how large the page can be**

3. The Screen object…

- **The screen object properties:**
  - **availHeight**
    - **Returns the height of the screen (excluding the Windows Taskbar)**
  - **availWidth**
    - **Returns the width of the screen (excluding the Windows Taskbar)**
  - **colorDepth**
    - **Returns the bit depth of the color palette for displaying images**

3. The Screen object…

- **Height**
  - **Returns the total height of the screen**
- **Width**
  - **Returns the total width of the screen**

3. The Screen object…

```
<head>
    <title> Screen Object </title>
    <script language="javascript">
        document.write("Available Height: " + screen.availHeight)
        document.write("<br>Available Width: " + screen.availWidth)
        document.write("<br>Height: ", screen.height)
        document.write("<br>Width: ", screen.width)
    </script>
</head>
```

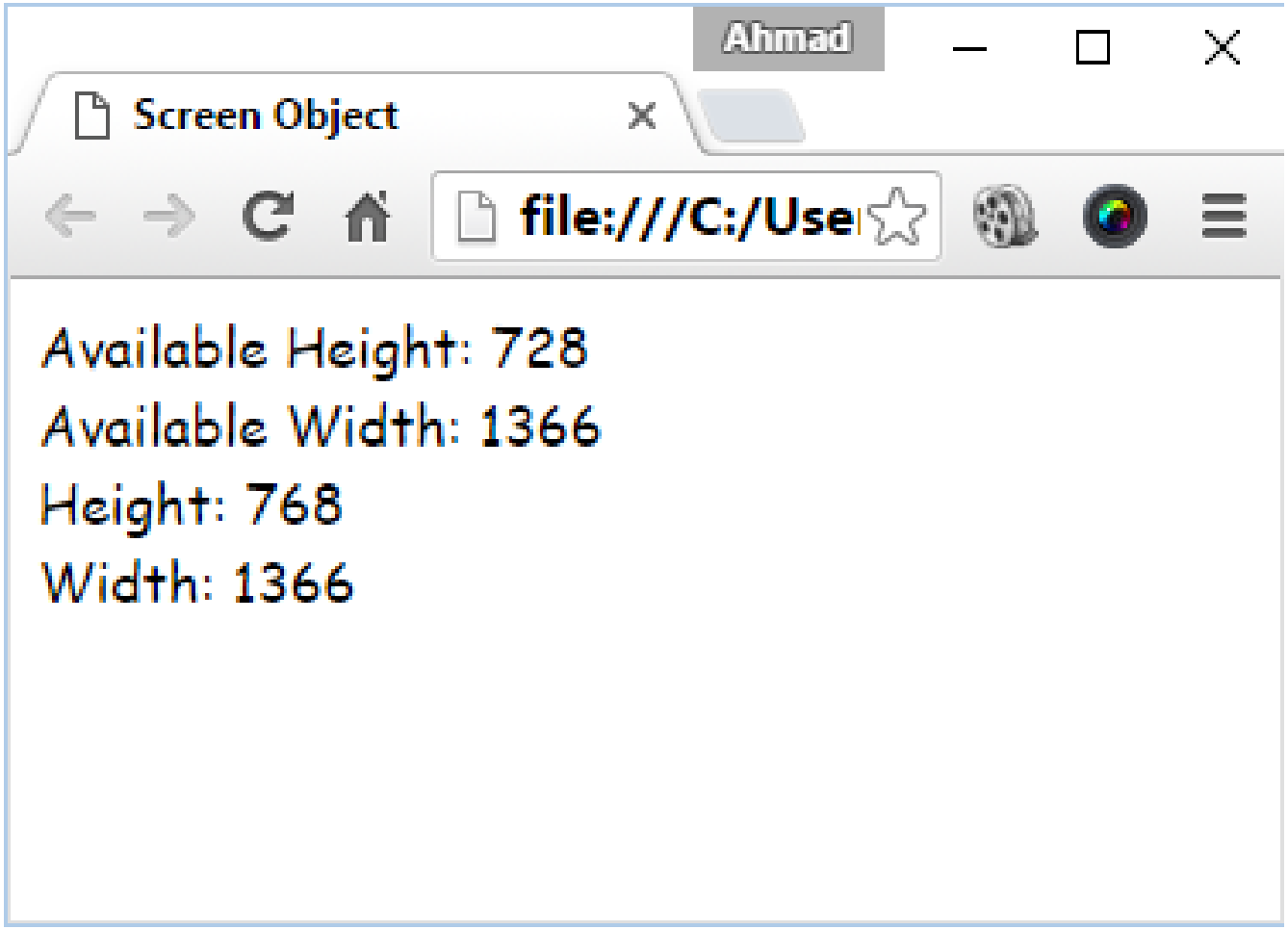
Finding Available Height

Finding Available Width

Finding Actual Height

Finding Actual Width

3. The Screen object…



4. Form Object

- The Form object represents an HTML form
- For each <form> tag in an HTML document, a Form object is created
- The browser crates a 'forms array' which keeps the number of form objects in the HTML program
- The first form object in the HTML file being held as array index [0], the second as index [1] and so on

4. Form Object…

- The ‘forms array’ also holds information about each element used within <form> and </form> tags
- elements array keeps information about form elements

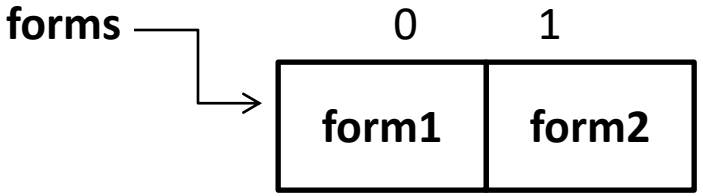
4. Form Object…

- **Create a Form Object**
- You can create a <form> element by using the **document.createElement() method:**

- **var x = document.createElement("FORM");**

4. Form Object…

```
<body>
        <form name="form1">
        </form>
        <form name="form2">
        </form>
</body>
```



4.1 Accessing Form Elements

- **Let's see, how to access the elements of the following form:**

```
<body>
        <form name="form1">
                <input type="text" name="name">
                <input type="text" name="email">
        </form>
</body>
```

4.1 Accessing Form Elements…

- **Accessing Name Element**
  - **document.forms[0].name.value**
  - **document.form1.elements[0].value**
- **Accessing Email Element**
  - **document.forms[0].email.value**
  - **document.form1.elements[1].value**

4.2 Setting Form Elements

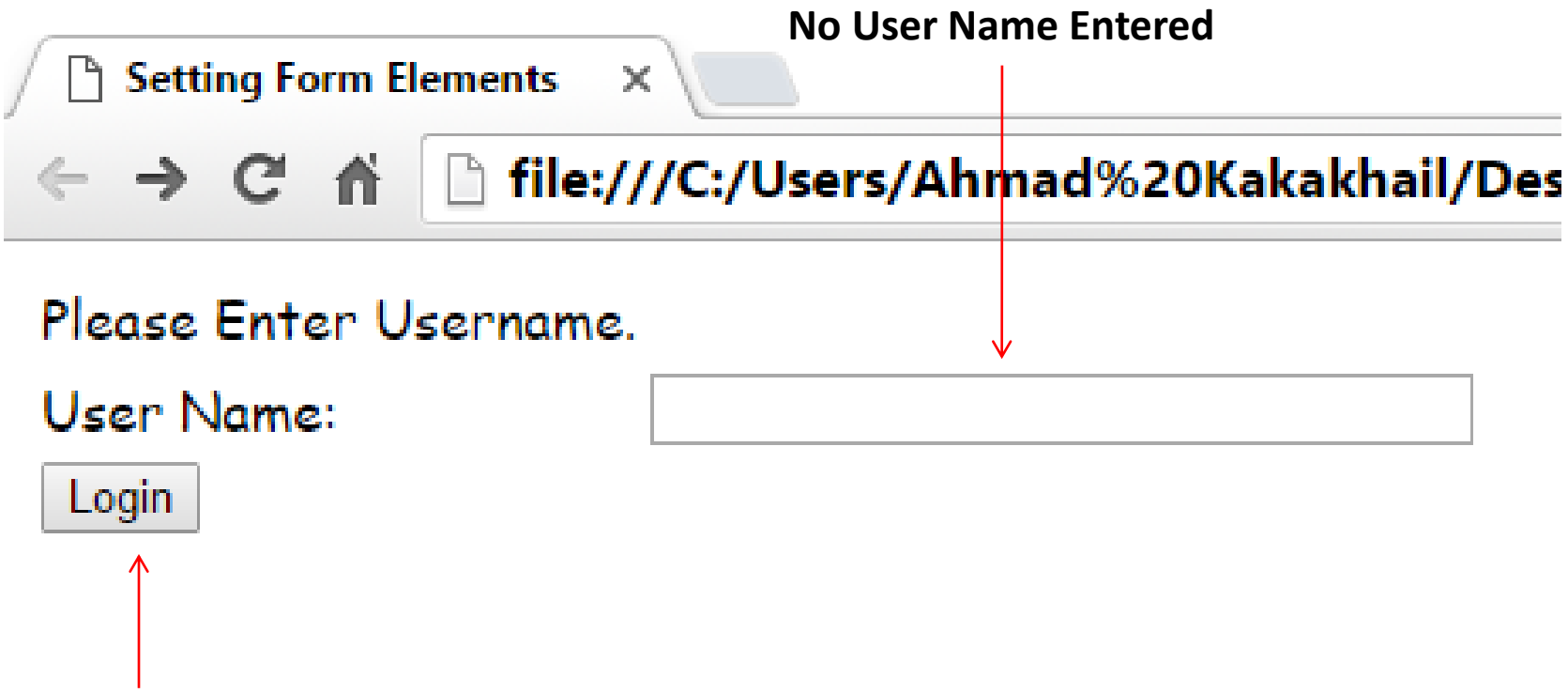
- **Let's see, how to set the elements of the following form**

```
<body>
        <form name="form1">
                <input type="text" name="name">
                <input type="text" name="email">
        </form>
</body>
```

4.2 Setting Form Elements…

- **Setting Name Element**
  - **document.forms[0].name.value = “Ali”**
  - **document.form1.elements[0].value = “Ali”**
- **Setting Email Element**
  - **document.forms[0].email.value = “ali@gmail.com”**
  - **document.form1.elements[1].value = “ali@gmail.com”**

4.2 Setting Form Elements…



4.2 Setting Form Elements…

Set Form Function Call

```
<form name="index" action="indexaction.php" onSubmit="setForm();" method="post">
    <table>
        <tr> <td>Please Enter Username.</td> </tr>
        <tr>
            <td>User Name: </td>
            <td><input name="username" type="text" size="30" maxlength="32"/></td>
        </tr>
        <tr> <td> <input type="Submit" name="Submit" value="Login" class="inputSubmit"> </td> </tr>
    </table>
</form>
```

4.2 Setting Form Elements…

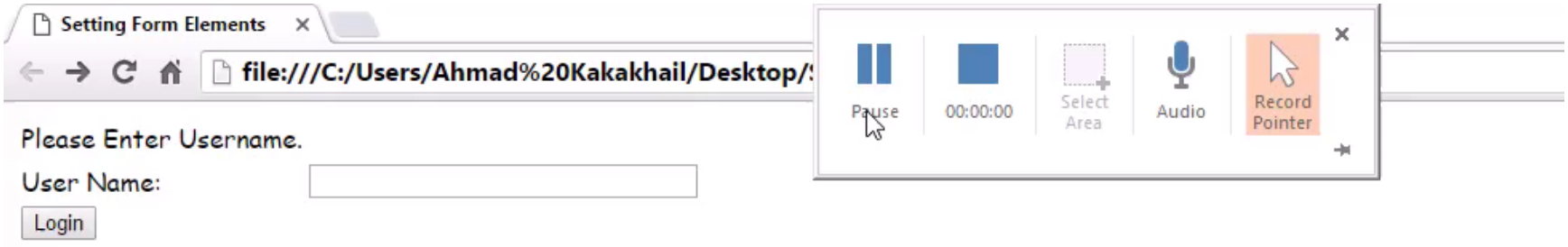
```
<script>
    function setForm()
    {
        var frm = document.index;
        if(frm.username.value=="")
        {
            frm.username.value = "Ahmad";
            alert('Username Replaced');
        }
    }
</script>
```

Getting Form Elements

Checking The Null Condition

Alert of User Name Replacement

4.2 Setting Form Elements…



4.3 Validating Form Data

- **JavaScript can be useful in validating the form submission that is:**
  - All required fields are filled with data?
    - **For Example:** If name is required, the textbox should not be empty on the time of submission.
  - All fields have valid data?
    - **For Example:** If email is required, it should not have invalid data like tra731, it must contain "@" and "."

4.3 Validating Form Data…

Validate Form Function Call

```
<body>
    <form name="index" action="indexaction.php" onSubmit="return validateForm();" method="post">
        <table>
            <tr> <td>LOGIN</td> </tr>
            <tr> <td>Please Enter Username and password.</td> </tr>
            <tr>
                <td>User Name: </td>
                <td><input name="username" type="text" size="30" maxlength="32"/></td>
            </tr>
            <tr>
                <td align="left" valign="top" class="gridRow1">Password:</td>
                <td ><input name="password" type="password" class="inputText" id="password" value=""size="30" maxlength="32"/></td>
            </tr>
            <tr> <td> <input type="Submit" name="Submit" value="Login" class="inputSubmit"> </td> </tr>
        </table>
    </form>
```

4.3 Validating Form Data…

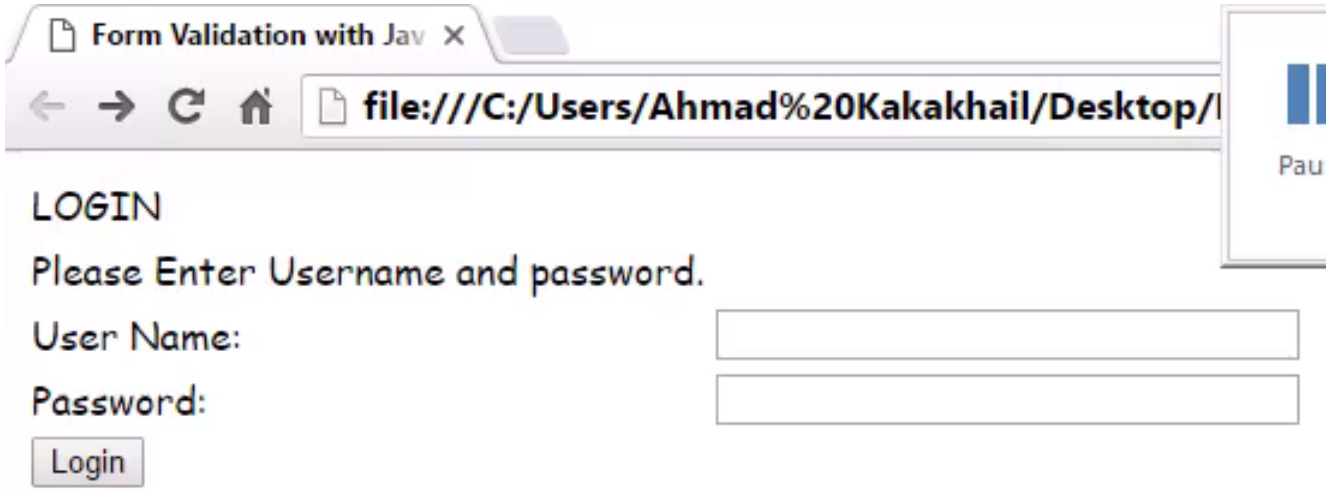
```
<script>
    function validateForm()
    {
        var frm = document.index;
        if(frm.username.value=="")
        {
            alert('Please Enter Username');
            frm.username.focus();
            return false;
        }
        else if(frm.password.value=="")
        {
            alert('Please Enter Password');
            frm.password.focus();
            return false;
        }
    }
</script>
```

Getting Form Elements

Checking For Emptiness

Alerts When The Field is Empty

4.3 Validating Form Data…



5.0 Data Validation

- **Data validation is the process of ensuring that computer input is clean, correct, and useful.**
- **Typical validation tasks are:**
- **has the user filled in all required fields?**
- **has the user entered a valid date?**
- **has the user entered text in a numeric field?**

5.0 Data Validation

- **Most often, the purpose of data validation is to ensure correct input to a computer application.**
- **Validation can be defined by many different methods, and deployed in many different ways.**
- **Server side validation is performed by a web server, after input has been sent to the server.**
- **Client side validation is performed by a web browser, before input is sent to a web server.**

Summary of Today's Lecture

- The history object
- The navigator object
- The user browser
- The browser Engine
- The browser platform
- The browser language
- The screen object
- The form object
  - Accessing from element
  - Setting form element

Summary of Today's Lecture

- **Validating form data**
- **DATA Validation**

THANK YOU