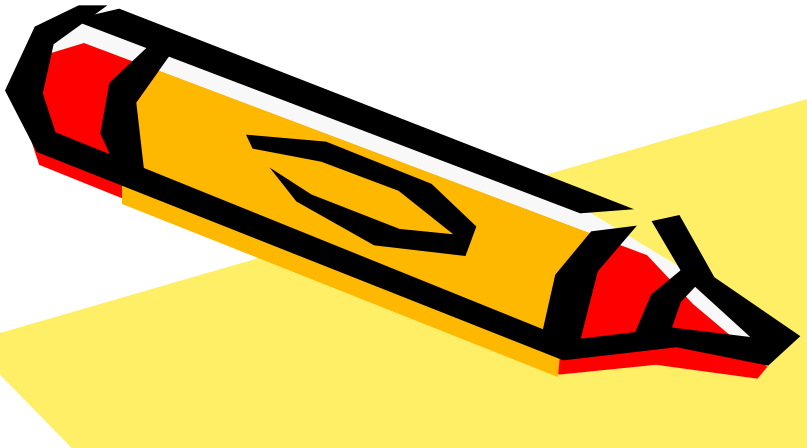
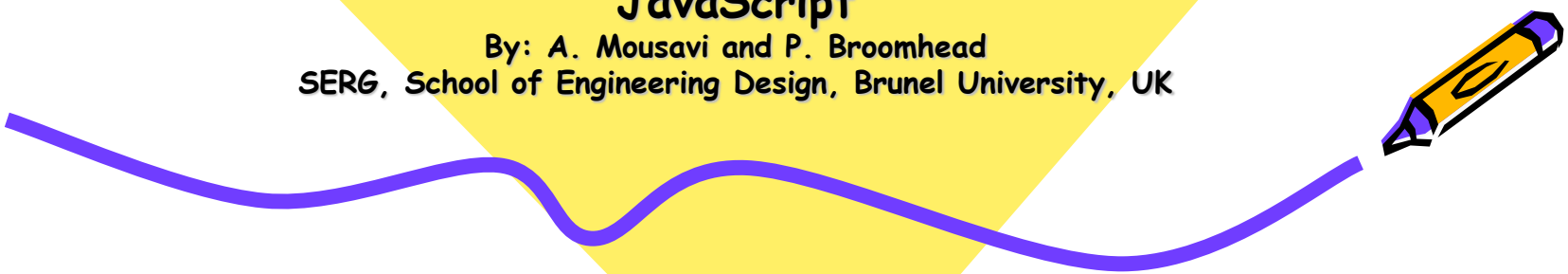


Programming for Digital Media EE1707

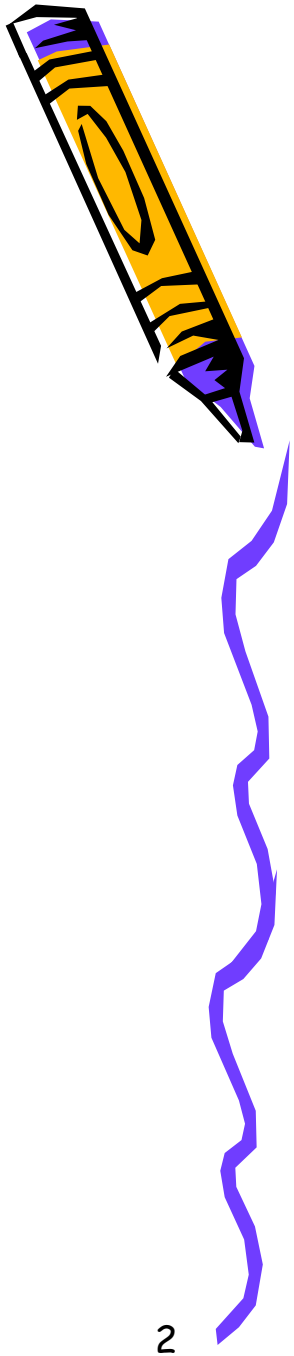
Lecture 5

JavaScript

By: A. Mousavi and P. Broomhead
SERG, School of Engineering Design, Brunel University, UK

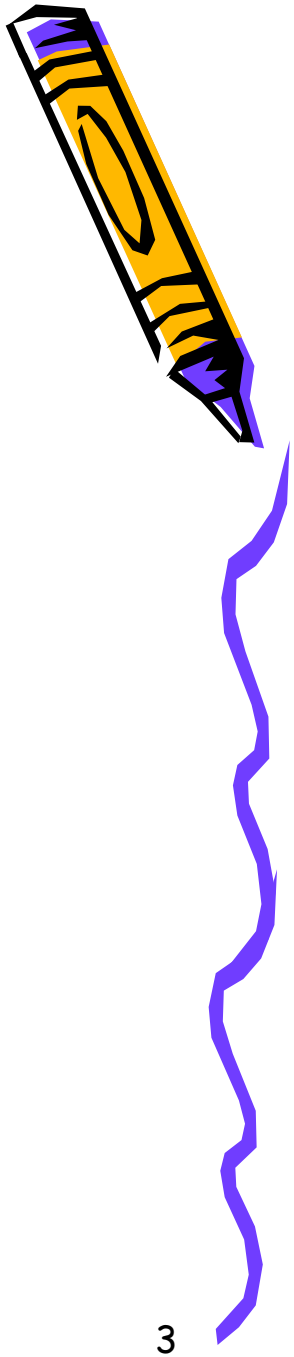


JavaScript and Forms



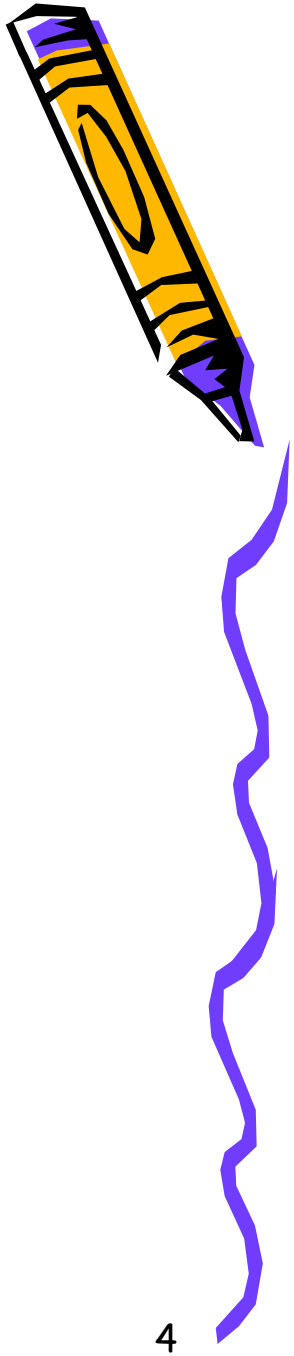
1. Interacting with fields in forms
2. Forms event handling
3. Validating forms

Forms



- Containers of multiple elements
- Allow user to input information
- Allow the programmer to check the sanity of information
- Allow manipulation and validation of data at client side

Form fields



- **Forms can be managed by JavaScript**
  - Uses *forms[ ]* array to access forms as part of the document object

document.forms[FormName].

- **Fields in the form can be accessed using the *elements[ ]* array**
  - A **name** associated with the form element

// to return value of a field in the form

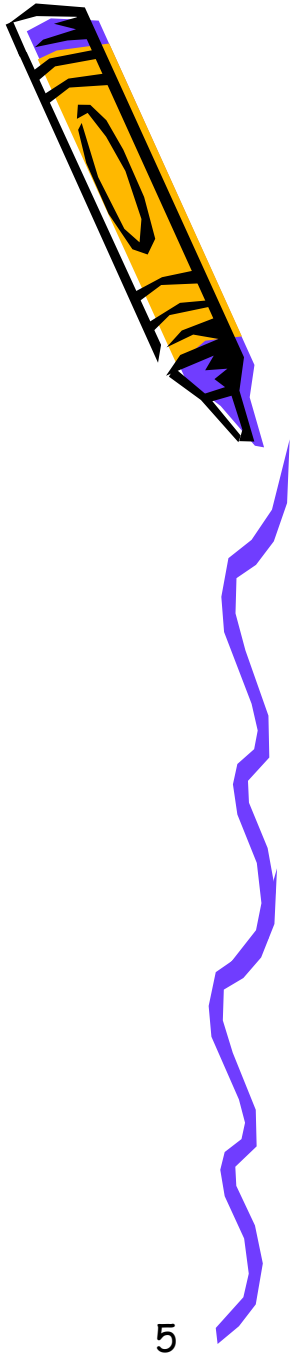
document.forms[formName].elements[fieldName].value

  - To return the name of the field on the form

// to return field name

document.forms[formName].elements[fieldName].name

Form elements



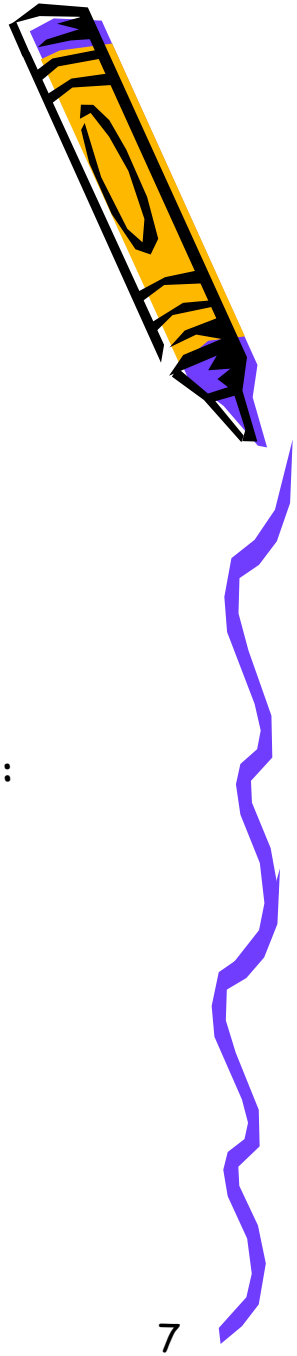
- **Input tag:**
  - Text Fields
  - Text Area
  - Radio Buttons (exclusive choice)
  - Check Boxes (multiple choice)
- **Drop Down List**
- **Action tag**
  - Submit button

Exercise 5-1

```
<html> <head><title> Form Example</title></head>
<body> <h2> Example of Form Elements</h2>
<form action="MyForm.html", name= "PunchForm", method = "get">
Enter Your Name:
<input type="text" name="firstname" />
<br />
<fieldset>
<legend>
Gender:
</legend>
<input type="radio" name="gender" value="male" /> Male
<input type="radio" name="gender" value="female" /> Female
</fieldset></br>
Choose your favorite Fruits:
</br>
```

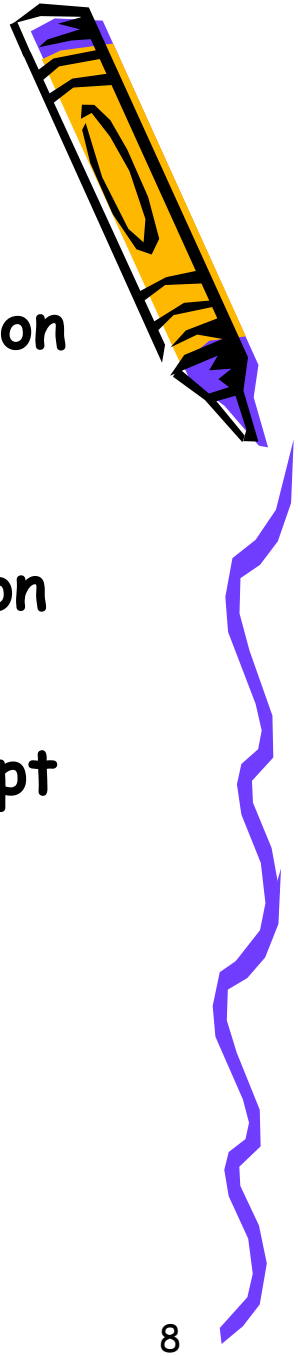
Continued next page

Example 6-1 Continued



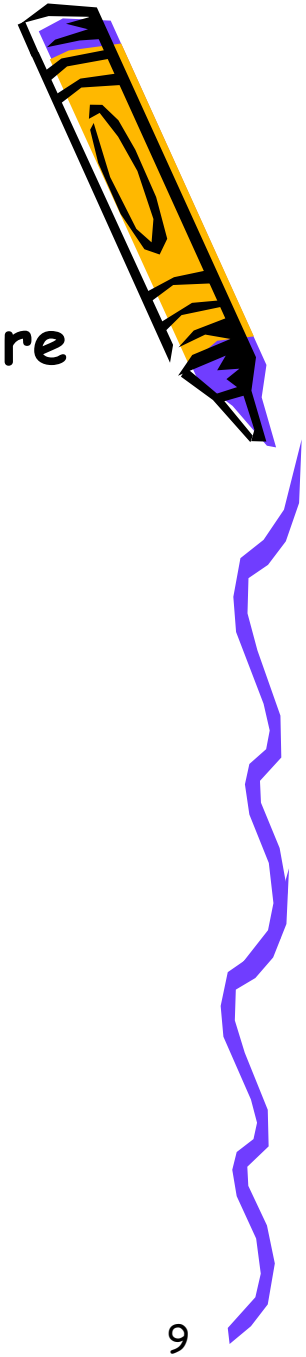
```
Apples:
<input type="checkbox" name="Fruits" value="Apples" />
<br />
Pears:
<input type="checkbox" name="Fruits" value="Pears" />
<br />
Oranges:
<input type="checkbox" name="vehicle" value="Oranges" />
</br>

Can you make a Punch with your favorite fruits? Please Type in your Instructions:
</br>

<textarea rows="10" cols="30"> </textarea>

</br>

<input type="submit" value="Submit" />


</form>
```

Form Validation



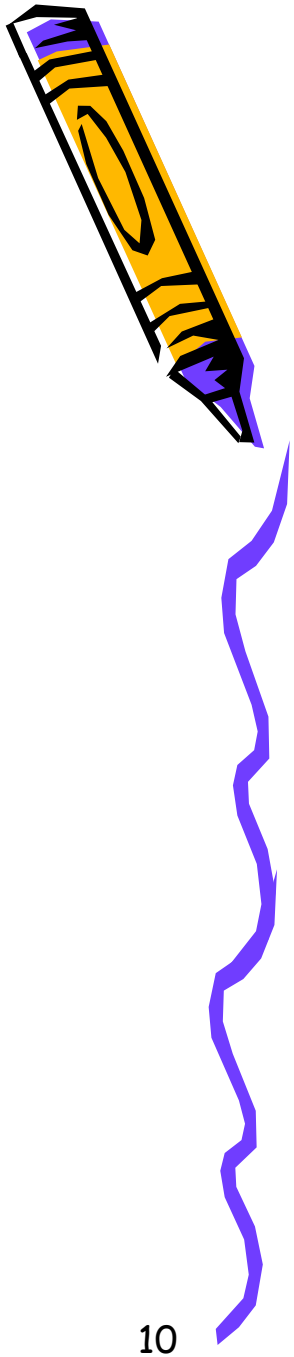
- Asking for information – you never know whether the user would provide the information in the correct format
- Client side validation and server side validation
- There are some advantages in using JavaScript to validate forms

Advantages to Validate Forms using JavaScript



- **Catch some of the mistakes in the data before transmission**
- **Improved user experience**
- **Quick response**
- **Reduced load on the server**

The Form object



- **The Form object allows for manipulation of HTML Forms**
  - **Methods:** *submit( )* and *reset( )*
  - **Properties:** *action, method, name, target*
  - **Events:** *onSubmit* and *onReset*

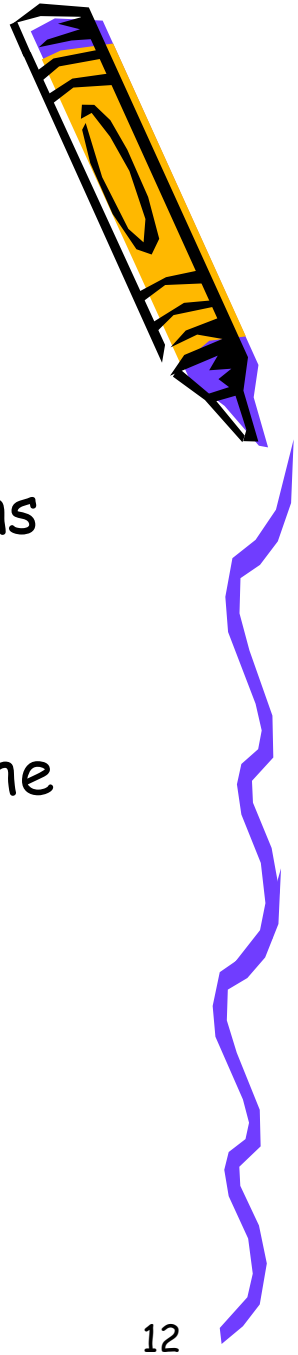
Example (4-5 form message, Designing with JavaScript, Second Edition, N. Heinle & B. Pena, O'Rielly Web Studio – list of references lecture 1)

```
<html> <head> <title> Send a Message Form </title> </head>
<body>
<h2> Send Me a Message</h2>
<form name = "MyForm" onSubmit= " return isReady(this);" method= "post"
                action= "messageform.html">

<p> <textarea name= "message" rows="10" cols="20" wrap="wrap" > </textarea></p>
<p> <input type="submit" value="Submit">
<input type="reset" value= "Reset"> </p>

</form>
</body>
</html>
```

In the form tag
Triggers the *isReady( )*
if isReady( ) returns true
The form is submitted

The event handlers trigger events by calling a function - note that this in the bracket means the current document (form)

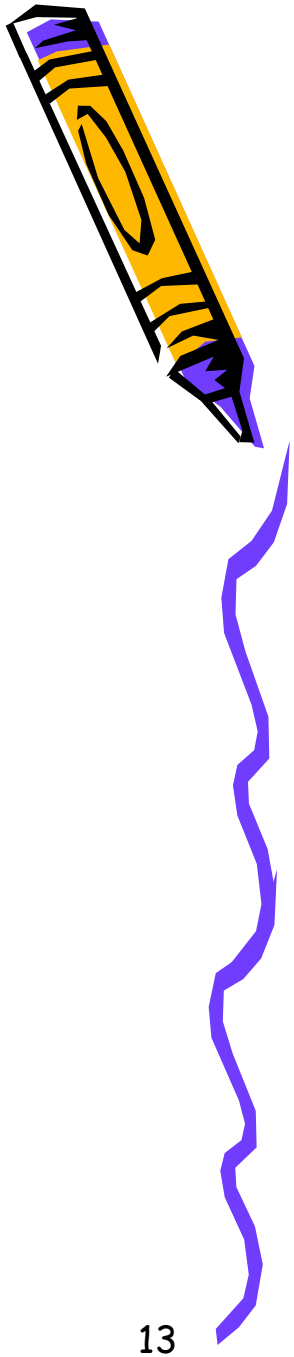
Form object methods



- The *submit ( ) :* submits a form - a validation function can call this method once the conditions are met

- The *reset( ):* resets the values in the form to the default

Form Validate Exercise

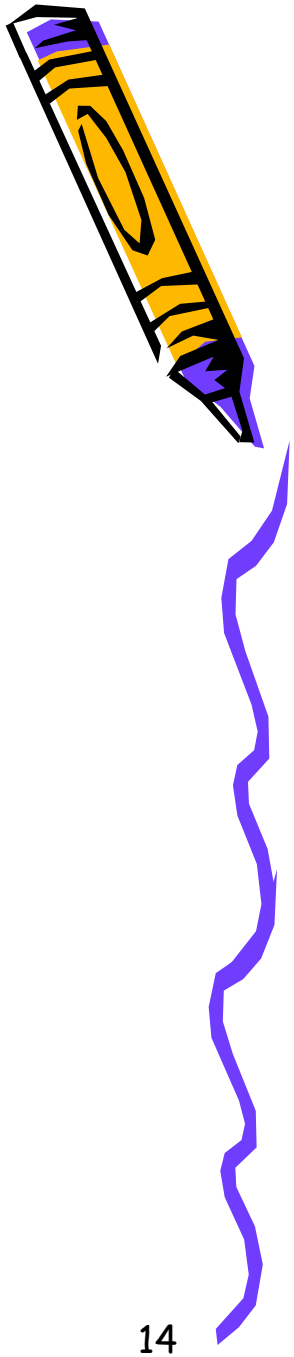


```
<html>
<head>
<script type="text/javascript">
function validate_email(field,alerttxt)
{
with (field)
  {
  apos=value.indexOf("@");
  dotpos=value.lastIndexOf(".");
  if (apos<1||dotpos-apos<2)
    {alert(alerttxt);return false;}
  else {return true;}
  }
}
```

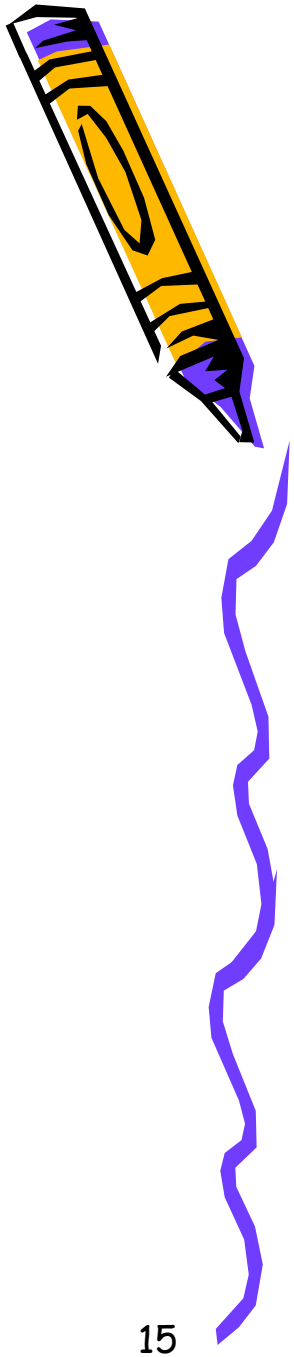
Continued on next slide

Example 5-2 cont.

```
function validate_form(thisform)
{
with (thisform)
  {
  if (validate_email(email,"Not a valid e-mail address!")==false)
    {email.focus();return false;}
  }
}
</script>
</head>

<body>
<form action="submit.htm" onsubmit="return validate_form(this);" method="post">
Email: <input type="text" name="email" size="30">
<p> Insert your Views</p>
<p> <textarea name="views" row="10" cols="40" warp="wrap"><textarea></p>
<input type="submit" value="Submit">
<input type="reset" value="Reset">
</form>
</body>

</html>
```



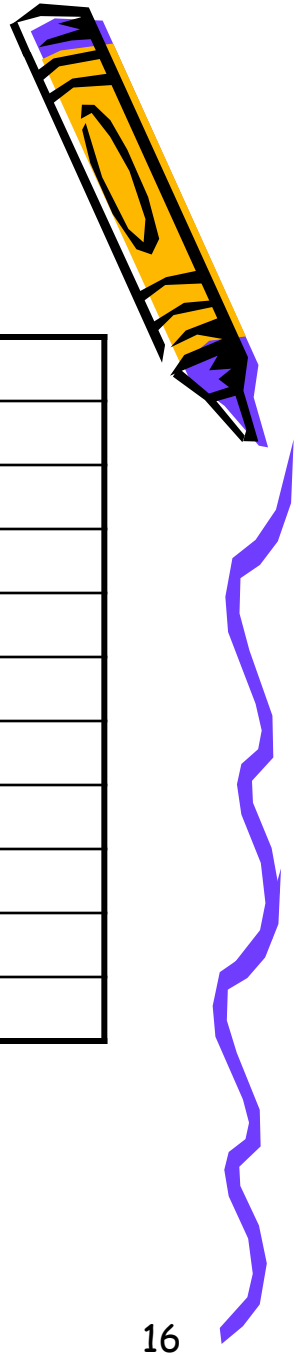
The Elements Array in Form



1. **Allows access to all elements in the form**

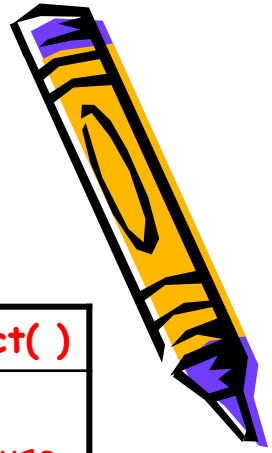
2. **The properties for each element are:**
   - *form, name, value,* and *type*
   - *checkbox* and *radio* objects
   - The *file* object that provides *FileUpload* feature
   - Text components

Form element properties



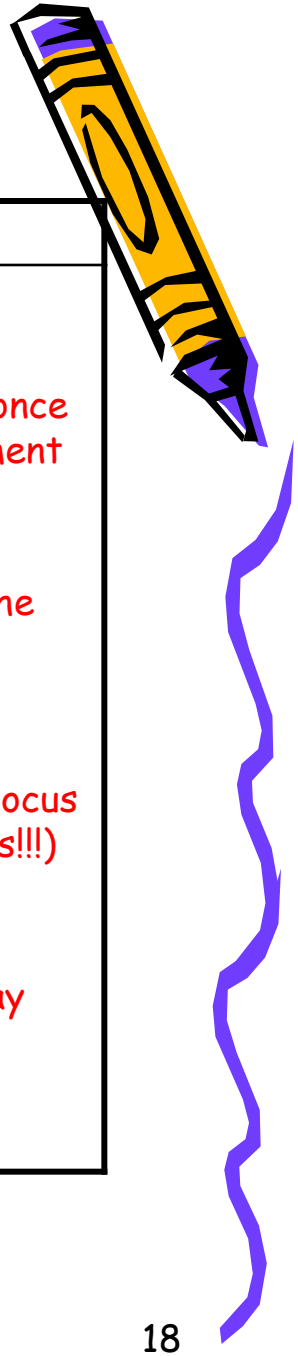
Element	Properties
Checkbox	form, name, type, value, checked, defaultChecked
FileUpload	form, name, type, value, defaultValue
Button	form, name, type, value
Radio	form, name, type, value, checked, defaultChecked
Reset	form, name, type, value
Select	form, name, type, length, options[ ], selectedIndex
Submit	form, name, type, value
Password	form, name, type, value, defaultValue
Text	form, name, type, value, defaultValue
Textarea	form, name, type, value, defaultValue

methods of form element



Element	Method	blur( ), click( ), focus( ), select( )
Checkbox	blur( ), click( ), focus( )	blur( ): removes keyboard or mouse focus from a form element focus( ): gives an element keyboard or mouse focus click( ): simulates mouse click on the corresponding form element select( ): highlights the text on the form element
Button	blur( ), click( ), focus( )	
FileUpload	blur( ), select( ), focus( )	
Radio	blur( ), click( ), focus( )	
Reset	blur( ), click( ), focus( )	
Select	blur( ), click( ), focus( )	
Submit	blur( ), click( ), focus( )	
Password	blur( ), select( ), focus( )	
Text	blur( ), select( ), focus( )	
Textarea	blur( ), select( ), focus( )	

Form element event handlers



Element	Event handler	functionality
Checkbox	onBlur, onClick, onFocus	onClick: Triggers an operation once clicking the mouse over an element onChange: is triggered once an element has been changed on the form onFocus: is triggered when the element on the form receives focus (NOTE: may cause infinite loops!!!) onBlur: triggered when a form element loses focus (NOTE: may cause infinite loops!!!)
FileUpload	onBlur, onClick, onFocus	
Button	onBlur, onClick, onFocus	
Radio	onBlur, onClick, onFocus	
Reset	onBlur, onClick, onFocus	
Select	onBlur, onClick, onFocus	
Submit	onBlur, onClick, onFocus	
Password	onBlur, onClick, onFocus	
Text	onBlur, onClick, onFocus	
Textarea	onBlur, onClick, onFocus	

More on event handlers and forms

- **Event handlers can be embedded in HTML**
  <input type="button" name "next" onClick="CalculateInterest( )">
- **Event handlers cannot call *document.write( )* to alter the current document**
- **Event handlers:**
  - Generate *alert( ), confirm( ), prompt( )* dialogs
  - Instigate new windows to open using *window.open( )*
  - Manipulate properties and variables
  - Can functions and methods
- **See examples 5-1 and 5-2**

Form validation process

1. Field content analysis

2. Form input value are retrieved as strings (NOTE: numerical values are also treated as strings)

3. *String* object allows for string content validation and HTML conversion

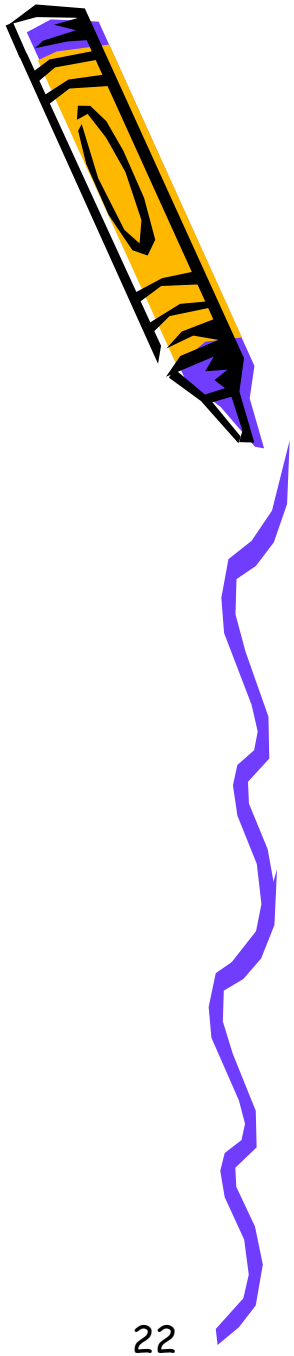
4. String functions allow for format conversions

String object

- **You can create String objects by:**
  1. *myString* = "THIS IS A STRING";
  2. myString = new String("THIS IS A STRING");

- **Useful methods to manipulate String object:**
  anchor( ), bold( ); fontcolor( ), fontsize( ), lastIndexOf( ), italics( ), link( ); toUpperCase( ), toLowerCase( ), …

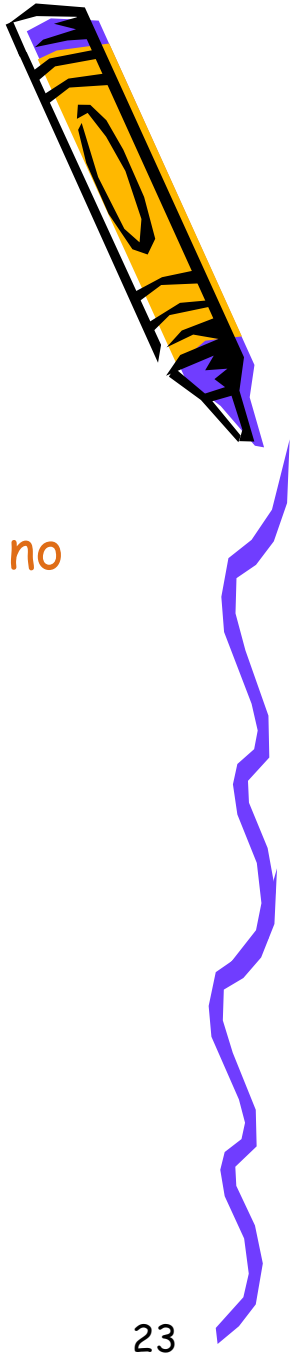
Converting Strings to Numerical Values



- **To convert strings to number use** *parseInt( )* and *parseFloat( )*
  - *Height = parseInt("135");*
  - *Height = parseFloat("5.8");*

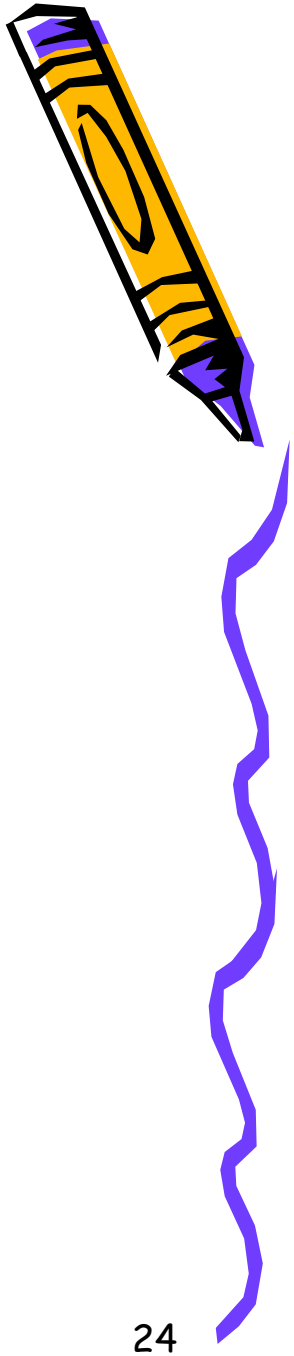
- **To check if user has entered valid numerical value use the** *isNaN( )* method

Accessing element values



- ***Select* object return:**
  - Options[ ] Array, length and *selectedIndex*
    - *selectedIndex* returns the *options[ ]* index of the selected item on the list and returns -1 if there is no items
- **The option[ ] objects return:**
  - text, value, length, index, selected, defaultSelected
  - Array indices start from 0

Accessing and Modifying Values in Select menus



- **Set the *text* or *value* of an option**

```
// replace the name of the option in a select menu to "Smooth Caramel"
theAssortedList = document.forms['FormName'].elements['SelectedName'];
the AssortedList.options[1].text='Smooth Caramel';
```

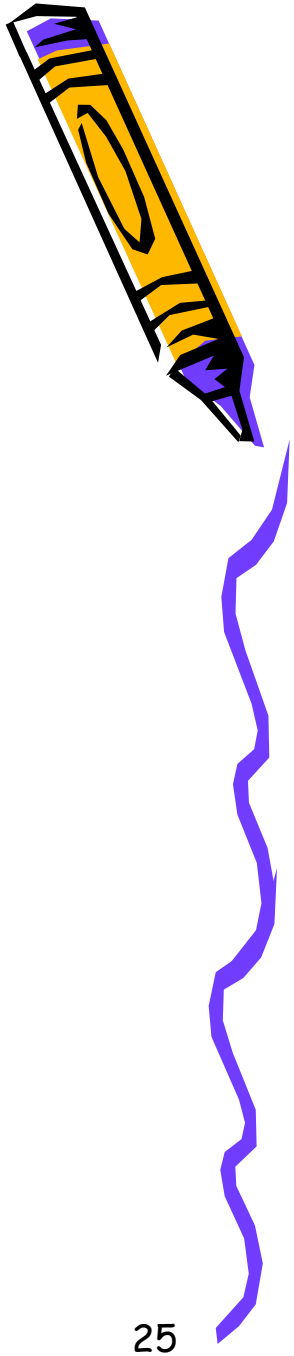
- *Increase the number of items* in *options[ ]* array

```
// add another item on the options [ ] array
var Listsize = theAssortedList.options.length;
theAssortedList.options.length = size+1;
theAssortedList.options[size].text='StrawberryJelly';
```

- To remove all from options

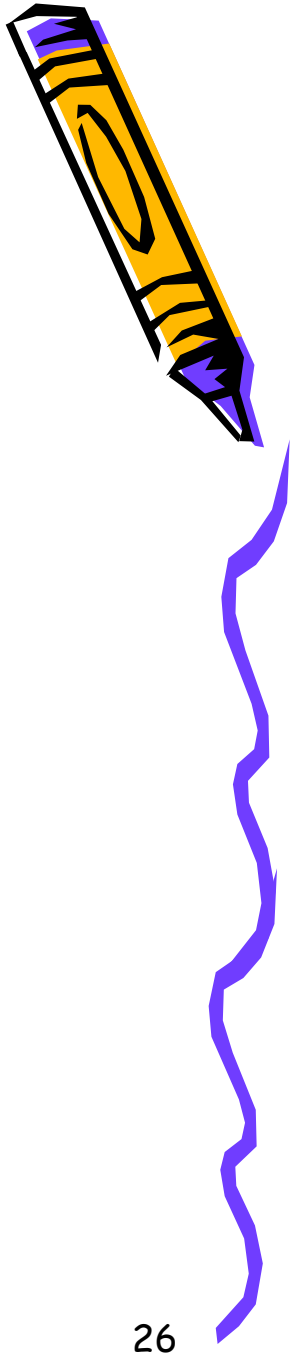
```
theAssorted.options.length = 0;
```

Example of Multiple Values from Select menu



```
<html><head><title>My Chocolate Bo </title>
<script language = "JavaScript">

Function theList(item) {
 if(item.selectedIndex == -1) {
     return;
     }

list= new String("  ");
for (var  count=0, count<item.options.length; count++) {
     if (item.options[count].selected == true) {
             list = list + item.options[count].text + "\n";
             }
        }
     }
Window.alert("You have Selected: \n" + list);
}
</script></head>
```

Example continued



```
<body>
<h2> List of Assorted Chocolates </h2>
<form>
<select size = "5"  name  "Chocolates"  multiple>
<option> Almond Butter </option>
<option> Dark Supreme </option>
<option> Walnut Heaven </option>
<option> Coffee Delight </option>
<option> Lost for Words </option>
</select>
<p>
<input  type ="button"  value="For You"
    onClick="theList(thisform.elements['Chocolates'])" >
</p>
</form>
</body>
</html>
```