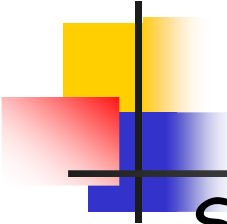


Lecture-2

- Javascript
 - Functions
 - Exceptions
 - Events



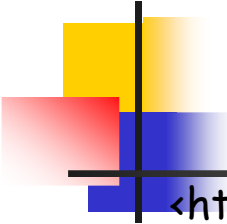
Javascript functions

Syntax

```
function <functionName> (params)
{
    // code
}
```

Note: Parameters do **NOT** have variable type.

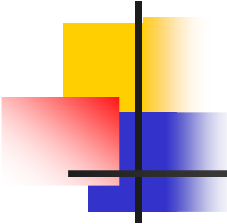
1. Recall: Function definition can be in
 1. <head> part of HTML file.
 2. <body> portion of HTML file
 3. An external file.
2. "return" value of the function is optional.



Functions - example1

```
<html>
  <head>
    <title> Example of a simple function </title>
    <script type="text/javascript">
      function factorial (input)
      {
        product = 1;
        for (i=1; i <= input; i++)
          product *= i;
        document.write ("factorial of i " + product);
      }
    </script>
  </head>

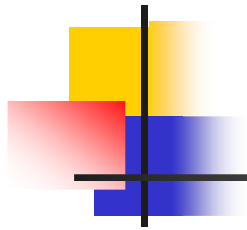
  <body>
    <h1> Example of a simple function </h1>
    <script type="text/javascript">
      factorial (9);
    </script>
  </body>
</html>
```



Functions - example2

```
<html>
  <head>
    <title>Browser Information example</title>
    <script type="text/javascript">
      function BrowserInfoFn()
      {
        var browser = navigator.appName;
        var version = navigator.appVersion;
        var ver = parseFloat (version);
        document.write ("Broswer: " + browser + " version: " +
                          version + " ver: " + ver + "<br />");
      }
    </script>
  </head>

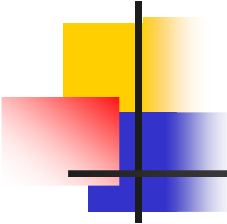
  <body>
    <h1>Browser Information example</h1>
    <script type="text/javascript">
      BrowserInfoFn();
    </script>
    <hr>
  </body>
</html>
```



Exceptions

- Syntax and usage
 - Similar to Java/C++ exception handling

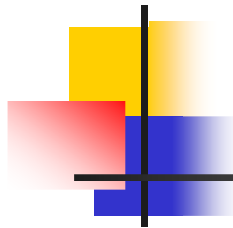
```
try
{
    // your code here
}
catch (excpn)
{
    // handle error
    // optional throw
}
```



Exceptions - example

```
<html>
<head>
  <title> Error handling example </title>
  <script type="text/javascript">
    function handleError( )
    {
      try
      {
        i = foo( ); // Calling an undefined function raises an exception
        return i;
      }
      catch (err)
      {
        alert ("Something was wrong");
        return ("Error!");
      }
    }
  </script>
</head>

<body>
  <script type="text/javascript">
    var i = handleError();
    document.write ("i: " + i);
  </script>
</body>
</html>
```



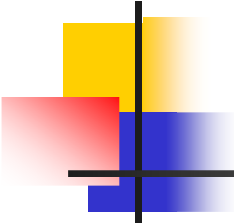
Throw

- Syntax
 - throw (exception)

Example

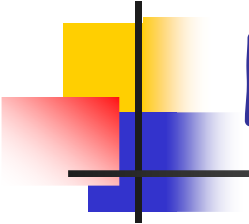
Try

```
{  
    // some code  
}  
catch (error)  
{  
    // handle error  
    throw (error2)  
}
```



Throw example

```
<html>
  <body>
    <script type="text/javascript">
      var x = prompt ("Enter a number between 0 and 10:", "");
      try {
        if (x > 10)
          throw "Err1";
        else if (x < 0)
          throw "Err2";
      }
      catch (er) {
        if (er == "Err1")
          alert ("Error! The value is too high");
        if (er == "Err2")
          alert ("Error! The value is too low");
      }
    </script>
  </body>
</html>
```

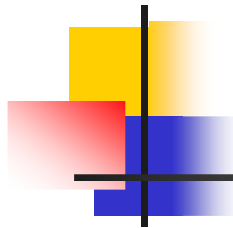



Handling errors - onerror

- onError - an **event** to handle errors.

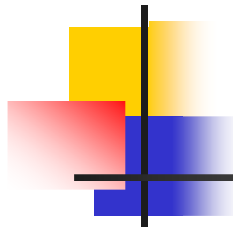
```
onerror=handleErr // Call handleErr on errors
```

```
function handleErr(msg,url,l)
// msg - error msg, url - current URL, l - line #
{
    //Handle the error here
    return true or false
}
```



Events

- Events in Javascript - "something" happening. Examples
 - Web page is loaded/unloaded
 - Mouse clicked/double-clicked
 - Mouse hovering over/out-of a region
 - Any key is pressed/released
 - An error has occurred
 - A "submit" button is pressed.
 - An element loses focus.
- Complete list of events are given at http://www.w3schools.com/jsref/jsref_events.asp

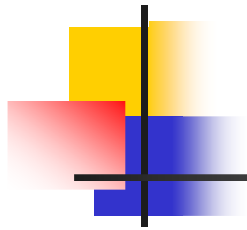


Events - onload()

Called (if defined) when a web page is loaded.

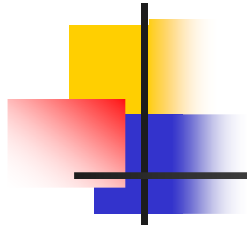
Simple Example

```
<html>
<head>
  <title> On Load event example </title>
  <script type="text/javascript">
    function onloadFn( )           // Function definition
    {
      alert ("Web page finished loading");
    }
  </script>
</head>
<body onload="onloadFn()">       // Call it when page is loaded
</body>
</html>
```



Events - onunload

- Opposite of onload
 - Called when
 - We go out of a web page or
 - A web page is re-loaded.
- Example
 - Same example as before
 - **Except** replace onload with onunload.



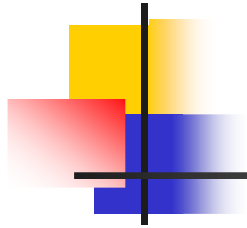
Events - onerror

- Recall our example of error handling

Example:

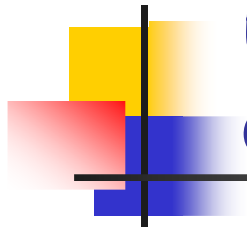
```
onerror=handleErr // Call handleErr on errors
```

```
function handleErr(msg,url,l)
// msg - error msg, url - current URL, l - line #
{
    //Handle the error here
    return true or false
}
```



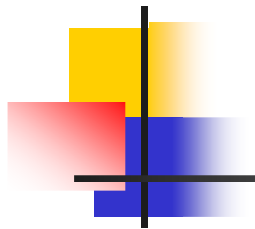
Events - onsubmit, onreset

- onsubmit - Event when a submit button is pressed.
 - Recollect our bank form example.
- onreset - event when a reset button is pressed.
 - Typically, used to cancel/reset the values of all fields.



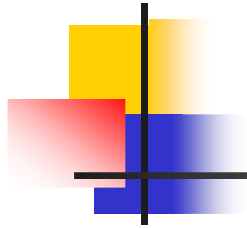
Events - onmouseup, onmousedown, onmouseover, onmouseout

- onmousedown - event when a mouse button is pressed down.
- onmouseup - event when a mouse button is pressed up.
- onmouseover - event when a mouse hovers over (a specific region).
- onmouseout - event when a mouse comes out (of a specific region).



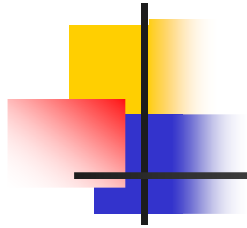
Events - onkeypress, onkeydown, onkeyup

- onkeypress - Event when a key is pressed
- onkeydown - Event when a key is pressed or held down
 - Similar to onkeypress
- onkeyup - Event when a key is released (after being pressed)



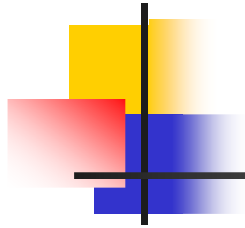
Events - onclick, ondblclick, onchange

- onclick - event when a button is clicked
- ondblclick - event when a button is double clicked
 - Try to avoid onclick when ondblclick is defined



Events - onfocus, onblur, onresize

- onfocus - event when an element gets focus
- onblur - event when an element loses focus
 - Opposite of onfocus



Events - onresize, onchange, onabort

- onresize - event when a browser window is resized (changed).
- onchange - event when the value of a field changes
- onabort - event when loading of an image is interrupted.