CS571: Programming Languages

What is JavaScript?

- JavaScript is a lightweight, interpreted programming language with object-oriented capabilities.
- Developed by Netscape
- JavaScript allows interactivity.
- Client-side language

Advantages of Javascript

- Less server interaction
  - Fast, no connection needed once loaded
  - Immediate feedback to the user
  - E.g. the code is executed when the user submits the form, and only if all the entries are valid they would be submitted to the Web Server.
- Increased interactivity
  - The user can interact with the webpage through the mouse and the keyboard.

Limitations with JavaScript

- JavaScript does not allow the reading or writing of files.
- JavaScript cannot be used for networking applications because there is no such support available.
- JavaScript does not have any multithreading or multiprocess capabilities.

Debugging JavaScript

- Debugging javascript
  - Developer tool in IE, Firefox, Chrome (press F12).
  - Safari:
    http://petewarden.com/2008/07/07/how-to-debug-ja/

A Simple Example (hello.html)

- JavaScript statements are placed within the <script>... </script> HTML tags in a web page.
- All statements end with ;
- Semicolon can be omitted if statements are each placed on a separate line.
- A simple example

```
<script language="javascript">
        document.write("Hello World!")
</script>
```

Comments

- Comments
  - // or <!-- : comment one line
  - /* and */: comment multiple lines

JavaScript in HTML (hello1.html)

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
    document.write("Hello World")
</script>
 <p>This is web page body </p>
</body>
</html>
```

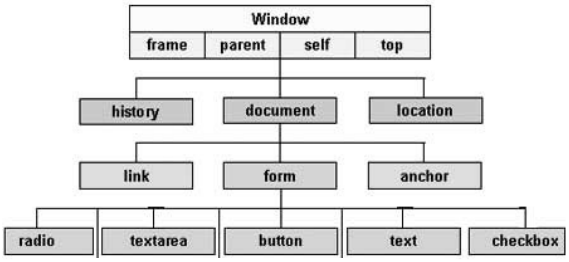

Types

- Number: eg. 123, 120.50 etc.
- String: e.g. "This text", 'this text' etc.
- Boolean: e.g. true, false.
- object

Document Object Model (DOM)

- Every web page resides inside a browser window which can be considered as an object.
- A Document object represents the HTML document that is displayed in that window.
- The way that document content is accessed and modified is called the Document Object Model, or DOM.

Object Hierarchy



Document Object Model (DOM)

- The name of each object is prefaced by the names of all the Objects that contain it.
- Each object has properties and methods
  e.g. window.alert(),
  window.document.write()

Variables

- Variables are declared using the var keyword

```
<script type="text/javascript">
    var m, name;   </script>
```

- JavaScript is case-sensitive.
  - E.g. time and TIME are different.
- Variables must begin with a letter or _
- A variable can hold a value of any data type and the type of a variable can change during the execution of a program

```
a = 5;              // now a number
a = "javascript";   // now a string
```

13

Operators

- Arithmetic operators: +, -, *, /, %(modulus), ++, --,
- Comparison operators: ==, !=, >, <, >=, <=
- Logic operators: &&(and), || (or), ! (not)

14

Control Structures

15

Conditional Statements (if.html)

- if statement
- if … else statement
- if … else if … statement

```
<script type="text/javascript">
    var book = "maths";
    if (book == "history" ){
     document.write("<b>History Book</b>");
  }else if (book == "maths" ){
      document.write("<b>Maths Book</b>");
  }else{
       document.write("<b>unknown Book</b>"); }
</script>
```

16

Conditional Statements (if.html)

- if statement
- if … else statement
- if … else if … statement

```
<script type="text/javascript">
    var book = "maths";
    if (book == "history" ){
     document.write("<b>History Book</b>");
  }else if (book == "maths" ){
      document.write("<b>Maths Book</b>");
  }else{
       document.write("<b>unknown Book</b>"); }
</script>
```

Output: maths book

17

Switch-Case Statement (switch1.html)

- Switch-case: evaluates an expression; the statements are executed based on the value of the expression.

```
<script type="text/javascript">
 var grade='A'; document.write("Entering switch block<br />");
 switch (grade) {
        case 'A': document.write("A<br />"); break;
        case 'B': document.write("B <br />"); break;
        case 'C': document.write("C<br />"); break;
        case 'F': document.write("Failed<br />"); break;
        default: document.write("Unknown grade<br />") }
 document.write("Exiting switch block");  </script>
```

18

Switch-Case Statement (switch1.html)

- Switch-case: evaluates an expression; the statements are executed based on the value of the expression.

```
<script type="text/javascript">
 var grade='A'; document.write("Entering switch block<br />");
 switch (grade) {
        case 'A': document.write("A<br />"); break;
        case 'B': document.write("B <br />"); break;
        case 'C': document.write("C<br />"); break;
        case 'F': document.write("Failed<br />"); break;
        default: document.write("Unknown grade<br />") }
 document.write("Exiting switch block");  </script>
```

Output: Entering switch block
A
Exiting switch block

19

Switch-Case Statement (switch.html)

- Switch-case statement evaluates an expression; the statements are executed based on the value of the expression.

```
<script type="text/javascript">
 var grade='A';
 document.write("Entering switch block<br />");
 switch (grade) {
        case 'A': document.write("A<br />");
        case 'B': document.write("B <br />");
        case 'C': document.write("C<br />");
        case 'F': document.write("Failed<br />");
        default: document.write("Unknown grade<br />")
}
 document.write("Exiting switch block");  </script>
```

20

Output

- Output:

Entering switch block
A
B
C
Fail
Unknown grade
Exiting switch block

21

While Loop (while.html)

- While: Loops when the boolean expression is true
- Example :

```
<script type="text/javascript">
   var count = 0;
   document.write("Starting Loop" + "<br />");
   while (count < 10){
        document.write("Count : " + count + "<br />");
        count++;
   }
document.write("Loop stopped!");
</script>
```

22

Output

Starting Loop
Count : 0
Count : 1
Count : 2
Count : 3
Count : 4
Count : 5
Count : 6
Count : 7
Count : 8
Count : 9
Loop stopped!

23

Do-While (dowhile.html)

- Do-while: similar to while except that the condition check happens at the end of the loop
- Example:

```
<script type="text/javascript">
 var count = 0;
 document.write("Starting Loop" + "<br />");
 do{
    document.write("Count : " + count + "<br />");
count++;
 }while (count < 0);
 document.write("Loop stopped!");
</script>
```

24

Output

```
Starting Loop
Count : 0
Loop stopped!
```


For Loops (for.html)

- for loop
  - Like C: for (initialization; condition; increment)
- Example:

```
<script type="text/javascript">
 var count;
 document.write("Starting Loop" + "<br />");
 for(count = 0; count < 10; count++){
     document.write("Count : " + count );
     document.write("<br />");
 }
 document.write("Loop stopped!");
 </script>
```


Output

```
Starting Loop
Count : 0
Count : 1
Count : 2
Count : 3
Count : 4
Count : 5
Count : 6
Count : 7
Count : 8
Count : 9
Loop stopped!
```


Loop Control (break.html)

- continue: ignore the current iteration
- break: terminates the loop.
- Example

```
<script language="javascript">
for (var i = 0; i < 5; i++) {
      if (i == 1) {continue; }
      else if(i == 3) {break; }
          else  {document.write(i + "<br />");}
} </script>
```


Loop Control (break.html)

- continue: ignore the current iteration
- break: terminates the loop.
- Example

```
<script language="javascript">
for (var i = 0; i < 5; i++) {
      if (i == 1) {continue; }
      else if(i == 3) {break; }
          else  {document.write(i + "<br />");}
} </script>
```

Output: 0
2

Functions (function.html)

- Functions are declared with the function keyword
- Example

```
<script type="text/javascript">
    function sayHello(name) {
     document.write("Hello " + name)
     }
    sayHello("alice")
</script>
```

Functions (function.html)

- Functions are declared with the function keyword
- Example

```
<script type="text/javascript">
    function sayHello(name) {
     document.write("Hello " + name)
     }
    sayHello("alice")
</script>
```

Output: Hello alice

Global vs Local Variables (local.html)

```
<script type="text/javascript">
  var x = "global"; // Declare a global variable function
  checkscope( ) {
      var x = "local"; // Declare a local variable
      document.write(x);
  }
  checkscope()
</script>
```


Global vs Local Variables (local.html)

```
<script type="text/javascript">
  var x = "global"; // Declare a global variable function
  checkscope( ) {
      var x = "local"; // Declare a local variable
      document.write(x);
  }
  checkscope()
</script>
```

Output: local

Javascript Objects

Javascript Object

- In JavaScript, almost everything is an object.
- Objects are variables. But objects can contain many values.

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```


Object (object.html)

- Create an object: new Object()

```
<script type="text/javascript">
    var book = new Object();
    book.subject = "Perl";
    book.author = "abc";
    document.write("Name is : " + book.subject + "<br>");
    document.write("Author is : " + book.author + "<br>");
</script>
```

Object (object.html)

- Create an object: new Object()

```
<script type="text/javascript">
    var book = new Object();
    book.subject = "Perl";
    book.author = “abc";
    document.write(“Name is : " + book.subject + "<br>");
    document.write(“Author is : " + book.author + "<br>");
</script>
```

Output:

Name is : Perl
Author is : abc

37

Object (object1.html)

- Define an object using the constructor function.

```
<script type="text/javascript">
function myObject(name) { this.name = name;}

var o = new myObject(“alice”);

document.write(“name: " + o.name + "<br>");
</script>
```

Output: name: alice

38

Defining Methods for an Object

```
function f() {
    …
}

function myObject() { this.f1= f;}

var o = new myObject();

o.f1();
```


39

Defining Methods For an Object (objfunc.html)

```
<script type="text/javascript">
    function addPrice(amount){ this.price = amount; }
    function book(title, author){
       this.title = title; this.author = author;
       this.addP=addPrice;}
    var myBook = new book("Perl", "Alice");
    myBook.addP(100);
    document.write("title: " + myBook.title + "<br>");
    document.write("author:" + myBook.author + "<br>");
    document.write("price: " + myBook.price + "<br>");
</script>
```

40

Defining Methods For an Object (objfunc.html)

```
<script type="text/javascript">
    function addPrice(amount){ this.price = amount; }
    function book(title, author){
       this.title = title; this.author = author;
       this.addP=addPrice;}
    var myBook = new book("Perl", "Alice");
    myBook.addP(100);
    document.write("title: " + myBook.title + "<br>");
    document.write("author:" + myBook.author + "<br>");
    document.write("price: " + myBook.price + "<br>");
</script>
```

Output: title: Perl
author:Alice
price: 100

41

Number Object (eval.html)

- Number object represents numerical date, either integers or floating-point numbers.

  var num = new Number(value);

- The browser automatically converts number literals to instances of the number class.

```
<script type="text/javascript">
    var x = new Number(10);
    var y = 20;
    document.write(eval("x * y") + "<br>");
</script>
```

Output: 200

eval(*string*): evaluate an expression ***string***.

42

Boolean (bool.html)

- Boolean object has two values : true, false.
- Creating a boolean object:

  var val = new Boolean(value);

  If value parameter is omitted, is 0, or false, then val has an initial value of false.
- The browser automatically converts true and false to instances of the boolean class.

```
var a,b;  a = new Boolean();  document.write(a);
if (a==true) {b = false; }else {b = true;} document.write(b);
```

43

Boolean (bool.html)

- Boolean object has two values : true, false.
- Creating a boolean object:

  var val = new Boolean(value);

  If value parameter is omitted or is 0, or false, then val has an initial value of false.
- The browser automatically converts true and false to instances of the boolean class.

```
var a,b;  a = new Boolean();  document.write(a);
if (a==true) {b = false; }else {b = true;} document.write(b);
```

output: false true

44

String

- A string is a sequence of characters, e.g.

  var t = new String("text");

  or

  var t = "text";

45

String Methods (string.html)

- charAt(): Returns the character at the specified index.

```
var x = "text";
document.write(x.charAt(1));
```

- slice(): Extracts a section of a string

  Slice(begin, end)

  slice() extracts up to but not including end

```
document.write(x.slice(1,3));
```

46

String Methods (string.html)

- charAt(): Returns the character at the specified index.

```
var x = "text";
document.write(x.charAt(1));  //the 2nd character
```

Output: e

- slice(): Extracts a section of a string

  Slice(begin, end)

  slice() extracts up to but not including end

```
document.write(x.slice(1,3));
```

47

String Methods (string.html)

- charAt(): Returns the character at the specified index.

```
var x = "text";
document.write(x.charAt(1));  //the 2nd character
```

Output: e

- slice(): Extracts a section of a string

  Slice(begin, end)

  slice() extracts up to but not including end

```
document.write(x.slice(1,3));  //characters at indexes 1 and 2
```

Output: ex

48

String Methods (string.html)

- slice(): Extracts a section of a string

Slice(begin, end)

slice() extracts up to but not including end

```
var x = "text";
document.write(x.slice(1));

var x = "text";
document.write(x.slice());
```

49

String Methods (string.html)

- slice(): Extracts a section of a string

Slice(begin, end)

slice() extracts up to but not including end

```
var x = "text";
document.write(x.slice(1));
```

Output: ext

```
var x = "text";
document.write(x.slice());
```

50

String Methods (string.html)

- slice(): Extracts a section of a string

Slice(begin, end)

slice() extracts up to but not including end

```
var x = "text";
document.write(x.slice(1));
```

Output: ext

```
var x = "text";
document.write(x.slice());
```

Output: text

51

String Methods (string.html)

- slice(): Extracts a section of a string

Slice(begin, end)

slice() extracts up to but not including end

```
var x = "text";
document.write(x.slice(7));
```

Output:

52

String Methods (string.html)

- +: Combines the text of two strings and returns a new string.

```
var x = "text";
 x += "book"
document.write(x);
```

Output: textbook

53

String Methods (string1.html)

- split(): Splits a String object into an array of strings by separating the string into substrings.

```
var s = "How are you doing today?";
var res = s.split(" ", 4);
document.write(res);
```


54

String Methods (string1.html)

- split(): Splits a String object into an array of strings by separating the string into substrings.

```
var s = "How are you doing today?";
var res = s.split(" ", 4);
document.write(res);
```

Output: How, are, you, doing

String Methods (lowcase.html)

- toLowerCase(): Returns the calling string value converted to lower case.

```
var x = "TEXt";
document.write(x.toLowerCase());
```

Output: text

- toUpperCase(): Returns the calling string value converted to uppercase.

Array (array.html)

- Index starts with 0

```
var fruits = [ "apple", "orange", "mango" ];
document.write(fruits[0] + "<br>");
document.write(fruits.length + "<br>");
document.write(fruits.indexOf("orange") + "<br>" );
document.write(fruits.pop() + "<br>");
fruits.push("watermelon");
document.write(fruits + "<br>");
```


Array (array.html)

- Index starts with 0

```
var fruits = [ "apple", "orange", "mango" ];
document.write(fruits[0] + "<br>");
document.write(fruits.length + "<br>");
document.write(fruits.indexOf("orange") + "<br>" );
document.write(fruits.pop() + "<br>");
fruits.push("watermelon");
document.write(fruits + "<br>");
```

output: apple
3
1
mango
apple,orange,watermelon

Array: Functions (array1.html)

```
var fruits = ["apple", "orange", "mango"];
document.write(fruits.reverse() + "<br>")
document.write(fruits.sort() + "<br>");
document.write(fruits.shift() + "<br>");
document.write(fruits.unshift("watermelon") + "<br>");
document.write(fruits + "<br>");
```

Output:

mango,orange,apple
apple,mango,orange
apple
3
watermelon,mango,orange

Date Object (date.html)

- Create a date object: **new Date();**
- Example:

```
var now = new Date();
document.write(now + "<br>"); //today's date and time
document.write(now.getDate() + "<br>");  //the day of month
                                         //(1-31)
document.write(now.getMonth() + "<br>"); //0-11
document.write(now.getDay() + "<br>");  //the day of week
                                        //(0-6)
document.write(now.getHours() + "<br>");  //the hour (0-23)
```

Date Object (date.html)

```
var now = new Date();
document.write(now + "<br>"); //today's date and time
document.write(now.getDate() + "<br>");  //the day of month
                                          //(1-31)
document.write(now.getMonth() + "<br>"); //0-11
document.write(now.getDay() + "<br>");  //the day of week
                                          //(0-6)
document.write(now.getHours() + "<br>");  //the hour (0-23)
```

```
Fri Jan 19 2018 23:31:49 GMT-0500 (EST)
19
0
5  //Friday
23
```


Exception Handling

Exception Handling (Cont.)

- Javascript uses try-catch-finally and throw to handle exception.
  - The try block must be followed by exactly one catch block, one finally block, or one of both

```
<script type="text/javascript">
  try { // Code that may throw an exception }
  catch ( e ) {// Code to run if an exception occurs }
  [finally {
    // Code that is always executed regardless of
    // an exception occurring
  }]
</script>
```

63

Example (exception.html)

```
<html> <head>
  <script type="text/javascript">
    function myFunc() {
      var a = 100;
      try {alert("Value of variable a is : " + a );}
      catch ( e ) {alert("Error: " + e.description );}
      finally {alert("Finally block will always execute!" );}}
  </script>
</head> <body>
  <p>Click the following to see the result:</p>
  <form>
    <input type="button" value="Click Me" onclick="myFunc();" />
  </form>
</body> </html>
```

64

The throw Statement (throw.html)

throw statement: raise your customized exceptions.

```
<html><head>
  <script type="text/javascript">
    function myFunc(){
      var a = 100; var b = 0;
      try{
        if ( b == 0 ){throw( "Divide by zero error." ); }
        else{var c = a / b;} }
      catch ( e ) {alert("Error: " + e ); }
    }
  </script>
</head> <body>
  <p>Click the following to see the result:</p>
  <form>
    <input type="button" value="Click Me" onclick="myFunc();" />
  </form>
</body> </html>
```

65

Dialog Box and Events

Alert Dialog Box (alert1.html)

- An alert dialog box is mostly used to give a warning message to the users.

```
<script type="text/javascript">
    alert("error");
</script>
```


Confirmation Dialog Box (confirm.html)

- A confirmation dialog box is used to take user's consent. It displays a dialog box with two buttons: OK and Cancel.
- If the user clicks on OK, confirm() will return true. If the user clicks Cancel, confirm() returns false.

```
<script type="text/javascript">
var retVal = confirm("Do you want to continue ?");
if( retVal == true ){
  alert("User wants to continue!"); return true; }
else{
  alert("User does not want to continue!"); return false;
}
</script>
```


Prompt Dialog Box (prompt.html)

- The prompt dialog box is used to pop-up a text box to get user input.
- The dialog box is displayed using prompt() which takes two parameters: a label and a string to display.
- This dialog box has two buttons: OK and Cancel. If the user clicks OK, prompt() returns the entered string. If the user clicks Cancel, prompt returns "".

```
<script type="text/javascript">
var retVal = prompt("Enter your name : ", "your name
here");
alert("You have entered : " + retVal );
</script>
```


Events

- JavaScript's interaction with HTML is handled through events.
- Events: clicking buttons, pressing keys, closing/resizing windows etc.

Event	Value	Description
onchange	script	Script runs when the element changes
onsubmit	script	Script runs when the form is submitted
onreset	script	Script runs when the form is reset
onselect	script	Script runs when the element is selected
onblur	script	Script runs when the element loses focus
onfocus	script	Script runs when the element gets focus
onkeydown	script	Script runs when key is pressed
onkeypress	script	Script runs when key is pressed and released
onkeyup	script	Script runs when key is released
onclick	script	Script runs when a mouse click
ondblclick	script	Script runs when a mouse double-click
onmousedown	script	Script runs when mouse button is pressed
onmousemove	script	Script runs when mouse pointer moves
onmouseout	script	Script runs when mouse pointer moves out of an element
onmouseover	script	Script runs when mouse pointer moves over an element
onmouseup	script	Script runs when mouse button is released

Onclick Event (alert.html)

```
<html>
<head>
<script type="text/javascript">
    function sayHello() { alert("Hello World") } </script>
</head>
<body>
<input type="button" onclick="sayHello()" value="Hello"
/>
</body>
</html>
```

Onsubmit Event (submit.html)

```
<SCRIPT TYPE="text/javascript">
  function TestDataCheck(){
      var number = document.testform.number.value;
      var returnval;
      if (number != "") returnval = true;
      else{ alert("please enter a number"); returnval = false; }
      return returnval;
} </SCRIPT>
<FORM ACTION="hello.html" NAME="testform"
      onSubmit="return TestDataCheck()" >
number: <INPUT TYPE=TEXT NAME="number"><BR>
<INPUT TYPE=SUBMIT VALUE="Submit"> </FORM>
```


Onsubmit Event (submit.html)

```
<SCRIPT TYPE="text/javascript">
  function TestDataCheck(){
      var number = document.testform.number.value;
      var returnval;
      if (number != "") returnval = true;
      else{ alert("please enter a number"); returnval = false; }
      return returnval;
} </SCRIPT>
<FORM ACTION="hello.html" NAME="testform"
      onSubmit="return TestDataCheck()" >
number: <INPUT TYPE=TEXT NAME="number"><BR>
<INPUT TYPE=SUBMIT VALUE="Submit"> </FORM>
```


Onmouseover Event

```
<html>
<head>
<script type="text/javascript">
    function over() {alert("Mouse Over"); }
</script>
</head>
<body>
<div onmouseover="over()" > <h2> This is inside the
division </h2> </div> </body> </html>
```


Onblur & onfocus (blur.html)

```
<html><body>
  Enter your name:
      <input type="text" id="myInput"
       onfocus="focusFunction()"
       onblur="blurFunction()"><script>
  function focusFunction(){
      document.getElementById("myInput").style.background
  = "yellow";}
   function blurFunction(){
      document.getElementById("myInput").style.background
  = "red";}
</script></body></html>
```


Onchange (onchange.html)

```
<html><body>
<p>Select a colorfrom the list.</p>
<select id="mySelect" onchange="myFunction()">
   <option value="rRed">Red
   <option value="Blue">Blue
   <option value="White">White
</select>
<p id="demo"></p><script>
function myFunction() {   var x =
   document.getElementById("mySelect").value;
   document.getElementById("demo").innerHTML = "You
   selected: " + x;}
</script></body></html>
```


Page Redirection (redirect.html)

```
<script type="text/javascript">
function Redirect() {
   window.location="http://blackboard.binghamton.edu";
}
document.write("You will be redirected to blackboard in 1
sec.");
setTimeout('Redirect()', 1000);
</script>
```

Page Redirect (Cont., redirect1.html)

```
<script type="text/javascript">
var browsername=navigator.appName;
document.write(browsername);
//Firefox, Chrome and Safari returns "Netscape"
if( browsername == "Netscape" ) {
    window.location="http://.binghamton.edu";
} else if (browsername =="Microsoft Internet Explorer") {
    window.location="http://blackboard.binghamton.edu"; }
else { window.location="http://www.binghamton.edu";
}
</script>
```


Page Printing (print.html)

```
<script type="text/javascript">
</script>
<form>
<input type="button" value="Print"
onclick="window.print()" /> </form>
```


Canvas

HTML Canvas

- Canvas is used to draw graphics on a web page
- Canvas consists of a drawable region defined in HTML code with height and width attributes.
- javascript can be used to draw graphics inside canvas.

Draw a Line Inside Canvas (line.html)

```
<!DOCTYPE html><html><body>
<canvas id="myCanvas" width="200" height="100"
  style="border:1px solid #d3d3d3;">
        Your browser does not support the HTML5 canvas tag.
</canvas>
<script>
        var c = document.getElementById("myCanvas");
        // two-dimensional rendering context
        var ctx = c.getContext("2d");
        ctx.moveTo(0,0);
        ctx.lineTo(200,100);
        ctx.stroke();
</script></body></html>
```

Draw Text Inside Canvas (text.html)

```
<!DOCTYPE html><html><body>
<canvas id="myCanvas" width="200" height="100"
  style="border:1px solid #d3d3d3;">
        Your browser does not support the HTML5 canvas tag.
</canvas>
<script>
        var c = document.getElementById("myCanvas");
        var ctx = c.getContext("2d");
        ctx.font = "30px Arial";
        ctx.fillText("Hello World",10,50);
</script></body></html>
```

Hello World

Draw Linear Gradient (gradient.html)

```
<!DOCTYPE html><html><body>
<canvas id="myCanvas" width="200" height="100"
 style="border:1px solid #d3d3d3;">
        Your browser does not support the HTML5 canvas tag.
</canvas>
<script>
        var c = document.getElementById("myCanvas");
        var ctx = c.getContext("2d");  // Create gradientvar
        grd = ctx.createLinearGradient(0,0,200,0);
        grd.addColorStop(0,"red");
        grd.addColorStop(1,"white");// Fill with gradient
        ctx.fillStyle = grd;ctx.fillRect(10,10,150,80);
</script></body></html>
```

Game Canvas

Game Canvas (canvas.html)

- Creating a game area

```
function startGame() {myGameArea.start();}

var myGameArea = {
    canvas : document.createElement("canvas"),
    start : function() {
        this.canvas.width = 480;
        this.canvas.height = 270;
        // two-dimensional rendering context
        this.context = this.canvas.getContext("2d");
        // insert <canvas> into DOM
        document.body.insertBefore(this.canvas,
document.body.childNodes[0]);
    }
}
```


Displaying a Squre (squire.html)

- Add a square

```
var myGamePiece;
function startGame() {
    myGameArea.start();
    myGamePiece = new component(30, 30, "red", 10, 120);
}

function component(width, height, color, x, y) {
    this.width = width;
    this.height = height;
    this.x = x;    this.y = y;
    ctx = myGameArea.context;
    //set the color to fill rectangle
    ctx.fillStyle = color;
    ctx.fillRect(this.x, this.y, this.width, this.height);
}
```


Moving a Square (movesquare.html)

```
var myGameArea = {
  canvas : document.createElement("canvas"),
  start : function() {
      this.canvas.width = 480;    this.canvas.height = 270;
      this.context = this.canvas.getContext("2d");
      document.body.insertBefore(this.canvas,
document.body.childNodes[0]);
      //run updateGameArea every 20 millisecond
      this.interval = setInterval(updateGameArea, 20);
  },
  clear : function() {
      this.context.clearRect(0, 0, this.canvas.width, this.canvas.height);
  } }

  function updateGameArea() {
      myGameArea.clear();
      myGamePiece.x += 1;
      myGamePiece.update();
  }
```


Moving a Squre (Cont)

```
function component(width, height, color, x, y) {
    this.width = width;
    this.height = height;
    this.x = x; this.y = y;
    this.update = function(){
        ctx = myGameArea.context;
        ctx.fillStyle = color;
        ctx.fillRect(this.x, this.y, this.width, this.height);}
}
```

Control The Movement of Square (ctrlsquare.html)

```
function component(width, height, color, x, y) {
    this.width = width; this.height = height;  this.speedX = 0; this.speedY = 0;
    this.x = x; this.y = y;
    this.update = function() {
        ctx = myGameArea.context;    ctx.fillStyle = color;
        ctx.fillRect(this.x, this.y, this.width, this.height);
    }
    this.newPos = function()
            {this.x += this.speedX;  this.y += this.speedY; }
}

function updateGameArea() {
    myGameArea.clear();
    myGamePiece.newPos();
    myGamePiece.update();
}

function moveup() {myGamePiece.speedY -= 1; }
function movedown() {myGamePiece.speedY += 1; }
function moveleft() { myGamePiece.speedX -= 1;}
function moveright() { myGamePiece.speedX += 1;}
```

91

Control the Movement of Square (Cont)

```
function clearmove() {
    myGamePiece.speedX = 0;
    myGamePiece.speedY = 0; }

<div style="text-align:center;width:480px;">
<button onmousedown="moveup()" onmouseup="clearmove()">
UP</button><br><br>

<button onmousedown="moveleft()"
onmouseup="clearmove()"> LEFT</button>

<button onmousedown="moveright()"
onmouseup="clearmove()"> RIGHT</button><br><br>

<button onmousedown="movedown()"
onmouseup="clearmove()">DOWN</button></div>
```

92

Keyboard as Controller (keysquare.html)

```
var myGameArea = {
    canvas : document.createElement("canvas"),
    start : function() {
        window.addEventListener('keydown', function (e) {
            myGameArea.key = e.keyCode;
        })
        window.addEventListener('keyup', function (e) {
            myGameArea.key = false;
        })
    }, ..... }

function updateGameArea() {
    myGameArea.clear();
    myGamePiece.speedX = 0;  myGamePiece.speedY = 0;
    if (myGameArea.key && myGameArea.key == 37) {myGamePiece.speedX = -1; }
    if (myGameArea.key && myGameArea.key == 39) {myGamePiece.speedX = 1; }
    if (myGameArea.key && myGameArea.key == 38) {myGamePiece.speedY = -1; }
    if (myGameArea.key && myGameArea.key == 40) {myGamePiece.speedY = 1; }
    myGamePiece.newPos();  myGamePiece.update();
}
```

93

Game Obstacles (obstacle.html)

```
var myGamePiece;    var myObstacle;
function startGame() {
    myGamePiece = new component(30, 30, "red", 10, 120);
    myObstacle  = new component(10, 200, "green", 300, 120);
    myGameArea.start();
}
var myGameArea = {
    canvas : document.createElement("canvas"),
    start : function() {
        this.canvas.width = 480;    this.canvas.height = 270;
        this.context = this.canvas.getContext("2d");
        document.body.insertBefore(this.canvas, document.body.childNodes[0]);
        this.interval = setInterval(updateGameArea, 20);    },
    clear : function() {
        this.context.clearRect(0, 0, this.canvas.width, this.canvas.height);    },
    stop : function() {      clearInterval(this.interval);    }}
```

94

Game Obstacles (Cont.)

```
function component(width, height, color, x, y) {
    this.width = width;    this.height = height;
    this.speedX = 0;    this.speedY = 0;    this.x = x;    this.y = y;
    this.update = function() {
        ctx = myGameArea.context;       ctx.fillStyle = color;
        ctx.fillRect(this.x, this.y, this.width, this.height);    }
    this.newPos = function() {
        this.x += this.speedX;       this.y += this.speedY;    }
    this.crashWith = function(otherobj) {
        var myleft = this.x;       var myright = this.x + (this.width);
        var mytop = this.y;       var mybottom = this.y + (this.height);
        var otherleft = otherobj.x;      var otherright = otherobj.x + (otherobj.width);
        var othertop = otherobj.y;      var otherbottom = otherobj.y + (otherobj.height);
        var crash = true;
        if ((mybottom < othertop) || (mytop > otherbottom) ||
        (myright < otherleft) || (myleft > otherright)) { crash = false; }       return crash;
}}
```

95

Game Obstacles (Cont.)

```
function updateGameArea() {
    if (myGamePiece.crashWith(myObstacle)) {  myGameArea.stop();    }
    else {  myGameArea.clear();    myObstacle.x -= 1;    myObstacle.update();
            myGamePiece.newPos();      myGamePiece.update();    }}
    function moveup() { myGamePiece.speedY = -1; }
    function movedown() {    myGamePiece.speedY = 1; }
    function moveleft() {    myGamePiece.speedX = -1; }
    function moveright() {    myGamePiece.speedX = 1; }
    function clearmove() {    myGamePiece.speedX = 0;     myGamePiece.speedY = 0; }
    <div style="text-align:center;width:480px;">
    <button onmousedown="moveup()" onmouseup="clearmove()"
ontouchstart="moveup()">UP</button><br><br>
    <button onmousedown="moveleft()" onmouseup="clearmove()"
ontouchstart="moveleft()">LEFT</button>
    <button onmousedown="moveright()" onmouseup="clearmove()"
ontouchstart="moveright()">RIGHT</button><br><br>
    <button onmousedown="movedown()" onmouseup="clearmove()"
ontouchstart="movedown()">DOWN</button></div>
```

96