PIC 40A

Lecture 10: JS: Wrapper objects, Input and Output, Control structures, random numbers

Objects in JS

In C++ we have classes, in JS we have OBJECTS.

We will not study how to make objects in this course, but we will use a lot of predefined objects.

Objects can have methods and properties.

Method is like a member function of a class and property is like a data variable stored by the class.

Objects in JS

Example:

```
document.write("Hello world!");
```

Here document is an object and write is a method.

We will see many more examples of properties and methods in weeks to come.

Recall

JavaScript has primitive data types

Number
String
Boolean

Each of these data types have corresponding wrapper objects: Number, String, Boolean.

Confused yet?

Wrapper objects

Wrapper objects are predefined objects that have useful methods and properties.

Lets say that I have a string variable

var name=”Fry”;

String variable by itself is just an ordinary variable that has no methods or properties. However it can be coerced to its corresponding wrapper String object and then it gains methods and properties.

Example

```
var name=”Fry”;
var len = name.length;
```

No parenthesis.
length is a property!

```
document.write(name + “ has “ + len + “ letters”);
```

Ordinary string variables like name do not have properties, but here name is coerced into its wrapper object.

String wrapper object

String wrapper object has the length property and the following methods:

Method	Parameters	What does it do
charAt	A number	returns the char at the specified position.
indexOf	One character string	Returns the location of the specified character.
substring	Two numbers. Start index is first, second is index of end+1.	Returns the substring of the string object from the first paramer to the second.
toLowerCase	None	Converts the string to lowercase
toUpperCase	None	Converts the string to oppercase

String examples

```
var name = “Philip J. Fry”;

var first_name = name.substring(0,6);

var letter = name.charAt(7); // J

var index = name.indexOf(“.”); // 8
```

Number wrapper object

Number object has method toString which we discussed earlier.

It has some properties as well: MAX_VALUE, MIN_VALUE, NaN, POSITIVE_INFINITY, NEGATIVE_INFINITY,PI.

These properties are for your reference only you will not be quizzed on them.

Math object

Math object is a predefined object.

All of Math objects methods are referenced through the Math object.

Example:

var value = Math.sin(x); // Here x is of type number

Some useful math methods include: sin, cos, floor, ceil, abs, exp, pow(x,y). See web for complete reference.

The Date object

On the web displaying information about date or writing code that depends on date or time information is common.

Date object is not a predefined object!

You have to invoke a constructor for the Date object. The constructor is predefined for you.

```
var today = new Date();
```

Date object methods

Method	Returns
`toLocaleString`	A string of Date information
`getDate`	The day of the month
`getMonth`	The month of the year in range 0 to 11
`getDay`	Day of the week in range 0 to 6
`getFullYear`	The year
`getTime`	Number of seconds since unix dawn of time (Jan. 1 1970)
`getHours`	Hour in the range 0 to 24
`getMinutes`	Minute in the range 0 to 59
`getSeconds`	Seconds in the range 0 to 59
`getMilliseconds`	Number in the range 0 to 999

Date Examples

```
<script type="text/javascript">

<!--
// Create Date object initialized to time right now
var today = new Date();

//Extract some info
var date_string = today.toLocaleString();
var day = today.getDay();
var month = today.getMonth();
var year = today.getFullYear();

//Write some stuff on the screen
document.write ("<h1>Welcome to my website!</h1> <h1> Today is: " + date_string + "</h1>");
document.write ("Day: " + day + "<br/>");
document.write ("Month: " + month + "<br/>");
document.write ("Year: " + year + "<br/>");

document.write ("How long does it take you to count to 10000000?<br/>");

//Get start time
var start = new Date();
//Loop!
for (var i = 0; i < 10000000; i++);
//Ok what is the time after the looping
var end = new Date();
//Difference is how long the loop took
var diff = end.getTime() - start.getTime();
//Write the result
document.write ("I can do it in " + diff + " milliseconds!!");

// -->
</script>
```

JavaScript Input and Output

```
document.write( ... );
```

Examples:

```
var a="Hello world!";
document.write(a);

document.write("Hello World");
```

In general

```
document.write(any_variable);
```

or

```
document.write(any_literal);
```

document.write(...); cont.

even more generally

```
document.write("The value of my_number is: ", my_number, "<br/>");
```

multiple values are concatenated.

But you can just concatenate yourself too!

```
document.write("The value of my_number is: " + my_number + "<br/>");
```

So it is not so useful to give multiple arguments after all.

alert

- An alert box is often used if you want to make sure information comes through to the user.
- When an alert box pops up, the user will have to click "OK" to proceed.

Syntax:

```
alert(some_string_paramater);
```

- some_string_paramater should not be XHTML
- So the string parameter could include \n but not

Example:

```
alert("The sum is:" + sum + "\n");
```

confirm

- The confirm function opens a dialog box with a message and an Ok button and a Cancel button.
- A confirm box is often used if you want the user to verify or accept something.
- When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed.
- Confirm returns either true if user presses Ok or false if user presses Cancel

Example:

```
var exit_question = confirm ("Are you sure you want to
exit?");
```

prompt

- Prompt function is a great way to have user input data
- The prompt function opens a dialogue box that contains a text box that can be used to get a string of input from the user
- Prompt box also has an Ok and Cancel buttons
- prompt function takes two arguments: The string used as a message in the box and a default value for the text box.
- Often empty string is used as default value

Example:

```
name = prompt("What is your name?", "");
```

Control Statements

- A program's **flow** is the order of execution of the program's statements.
- **Control statements** determine flow based on values of **control expressions** or **conditions.**

Examples of control statements:

– selection statements
- if-then, if-then-else, switch

– loop statements
- while, do-while, for, for-in

Control expressions can be

- primitive values
- relational expressions
- function calls
- object references
- compound expressions formed with any of the above by using logical operators such as `!` `&&` `||`
- Anything that is or can be coerced to a Boolean primitive value

Comparison Operators

Comparison operators are used in logical statements to determine equality or difference between variables or values.

Given that `x=5`, the table below explains the comparison operators:

Operator	Description	Example
==	is equal to	x==8 is false
===	is exactly equal to (value and type)	x===5 is true x==="5" is false
!=	is not equal	x!=8 is true
>	is greater than	x>8 is false
<	is less than	x<8 is true
>=	is greater than or equal to	x>=8 is false
<=	is less than or equal to	x<=8 is true

How are comparison operators used?

They are used to form boolean expressions which can be used as control expressions.

Example:

```
if (my_number > your_number)
    document.write("Sorry, you lose.");
```

Logical Operators

Logical operators are used to determine the logic between variables or values.

Given that **x=6 and y=3**, the table below explains the logical operators:

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x==5 || y==5) is false
!	not	!(x==y) is true

if Statement

```
if (condition)
  {
  // code to be executed if condition is true
  }
```

Example:

```
<script type="text/javascript">
var weather="";
weather = prompt("How is the weather?");
if (weather=="raining")
   alert("You better bring an umbrella.");
</script>
```

For compound statements use `if(...){ }`

For single statement `if(...) statement;` is ok

Example

Change the style sheet for the website depending on time of day.

```
var now = new Date().getHours();
if (0 <= now && now < 5)
     document.write("<link rel='stylesheet' href='night.css' type='text/css'>");

if (5 <= now && now < 11)
     document.write("<link rel='stylesheet' href='morning.css' type='text/css'>");

if (11 <= now && now < 16)
     document.write("<link rel='stylesheet' href='day.css' type='text/css'>");

if (16 <= now && now < 22)
     document.write("<link rel='stylesheet' href='evening.css' type='text/css'>");

if (22 <= now && now <= 24)
     document.write("<link rel='stylesheet' href='night.css' type='text/css'>");
```

if … else

```
if (condition)
  {
    // code to be executed if condition is true
  }
else
  {
    // code to be executed if condition is not true
  }
```

Example:

```
var d = new Date();
var time = d.getHours();

if (time < 10)
  {
  document.write("Good morning!");
  }
else
  {
  document.write("Good day!");
  }
```

if … else if ...else

```
if (condition1)
  {
    // code to be executed if condition1 is true
  }
else if (condition2)
  {
    // code to be executed if condition1 is not true and condition2 is true
  }


...

else
  {
    // code to be executed if none of the conditions above are not true
  }
```

Example

```
var score = -1;

score = prompt("Enter your score");

if ( score >= 90 )
   alert( "Your grade is A.\n Good work!");
else if ( score >= 80 )
   alert("Your grade is B.");
else if ( score >= 70 )
   alert("Your grade is C.");
else if (score >= 60 )
   alert("Your grade is D.");
else
   alert("You fail.\n See you again next quarter!");
```

while loop

The while loop loops through a block of code while a specified condition is true.

```
while (condition)
  {
    // code to be executed
  }
```

Example:

```
<body>
    <script type="text/javascript">
        var i=1;
        while(i<=6)
        {
            document.write("<h" + i + ">" + "This is header &lt;h" +
                                i + "&gt;" + "</h" + i + ">");
            i++;
        }
    </script>
</body>
```

for loop

The for loop is used when you know in advance how many times the script should run.

```
for (var i = startvalue; i <= endvalue; i += increment)
{
    // code to be executed
}
```

See example

How are comparison operators used?

They are used to form boolean expressions which can be used as control expressions.

Example:

```
if (my_number > your_number)
    document.write("Sorry, you lose.");
```

How are comparison operators used?

They are used to form boolean expressions which can be used as control expressions.

Example:

```
if (my_number > your_number)
    document.write("Sorry, you lose.");
```

do-while

JavaScript has a do-while loop that works exactly like C++ do-while.

Example:

```
var i=0;
do
{
 //stuff
 i+=2;
}while(i<20);
```

Random numbers

To get a random number in the range 0.0 up to but not including 1.0 we can write:

```
var my_random_number = Math.random();
```

To get numbers in the range we want we have to use combination of multiplication addition and floor math functions.

Example: Simulate a dice roll.

```
var dice = Math.floor( 1 + Math.random() * 6 );
```

Explanation: First we get a random number in the range 0 up to but not including 6 then we add one so we have a random number in the range 1 up to but not including 7. Then we floor it so we have an integer in the range 1 to 6 inclusive.

Is the random number generator seeded?

Yes it is seeded automatically.

General formula for integers

```
var number = Math.floor ( a + Math.random() * (b-a+1) );
```

This gives an integer in the range [a,b].

The scaling factor is b and the shifting value is a.