

**BIT 2<sup>nd</sup> Year**  
**Semester 3**  
**IT 3505**

**Web Application Development II**

**Fundamentals of Asynchronous  
JavaScript and XML (AJAX) –  
Part 2**



# JavaScript Objects

# JavaScript Objects

- In JavaScript, all values, except primitive values, are objects.
- A JavaScript object is an unordered collection of named items.
- The named item of an object can be a
  - Primitive value (called a property of the object).
  - function definition (called an object method).
  - Another object.
- JavaScript objects are specified by using the following syntax.  
    {name1:item1,name2:item2,.....,namex:itemx}
  - The individual values are separated by commas.
  - Each value is a pair of items separated by a colon.
  - In the namex:itemx pair the first part provides a symbolic name for a item given in the second part.

# JavaScript Objects .....

Example:

```
var student = {name:"saman",  
               age:20,  
               getdata: function(){  
                           return this.name+"  
"+this.age;}  
               };
```

The value of the **student** variable is a object with two properties (with the names **name** and **age**) and a single anonymous method.

# Constructing objects

- Objects can be constructed by
  - Specifying attributes and methods inside curly brackets.
  - Executing the command `new Object()` and then assigning items to the newly created object;  
Example :  

```
var obj = new Object();  
obj.name = "saman";  
obj.age = 20;  
obj.getdata = function(){ return this.name+"  
"+this.age;};
```
  - Creating an **object type** and then instantiate an object of that type.

# Creating an object type

```
function student(name,age){  
    this.name = name;  
    this.age = age;  
    this.getdata = function(){ return this.name+"  
"+this.age;};  
};  
var astudent = new student("saman",20); //Object  
instantiation
```

The function **student** is called an **object constructor**.

# this keyword

- When the keyword **this** is used in an object it indicates the object itself.

# Accessing object properties

- An object property can be accessed by using any of the following syntactic structures.
  - `objectName.property`
  - `objectName["property"]`
  - `objectName[expression]`
    - The expression must be evaluated to a property name



# Accessing object properties.....

Example:

```
var student = {name:"saman",  
               age:20,  
               };  
alert(student.name);  
alert(student["name"]);
```

# Looping through properties and methods of an object

The JavaScript construct for `..in` can be used to loop through all properties and methods of an object.

Syntax:

```
for (variable in object){  
    code to be executed  
}
```

# Looping through properties and methods of an object ....

Example :

```
var student = {name:"saman",  
               age:20,  
               };
```

```
for(property in student){  
    alert("Property name :"+property+ " value  
="+student[property]);  
}
```

# Adding a new property/method to an object ..

- A new property/method can be added to an object by assigning the new property/object with a value/definition.

Example:

```
var student = {  
    name : "Saman",  
    age : 20,  
    getdata : function(){ return this.name+" "+this.age;}  
};
```

```
student.sex = "male";
```

# Deleting a property/method from an object ..

- The **delete** keyword can be used to delete a property/method from an object. This will remove both property/method and its value/definition from the object.

Example:

```
var student = {  
    name : "Saman",  
    age : 20,  
    getdata : function(){ return this.name+" "+this.age;}  
};
```

```
delete student.age ;
```

# JavaScript Object Methods

- JavaScript object methods define actions that can be performed on objects.
- Syntax for defining object methods.

**methodName : function(){ method action}**

Object methods can be invoked by using the following syntax.

**objectName.methodName()**