Lesson Proper for Week 14

JavaScript is an Object Oriented Programming (OOP) language. A programming language can be called object-oriented if it provides four basic capabilities to developers:

¦ **Encapsulation** − the capability to store related information, whether data or methods, together in an object.

¦ **Aggregation** − the capability to store one object inside another object.

¦ **Inheritance** − the capability of a class to rely upon another class (or number of classes) for some of its properties and methods.

¦ **Polymorphism** − the capability to write one function or method that works in a variety of different ways.

Objects are composed of attributes. If an attribute contains a function, it is considered to be a method of the object, otherwise the attribute is considered a property.

Object Properties

Object properties can be any of the three primitive data types, or any of the abstract data types, such as another object. Object properties are usually variables that are used internally in the object's methods, but can also be globally visible variables that are used throughout the page. Properties are the values associated with an object.

The syntax for adding a property to an object is:

objectName.objectProperty = propertyValue;

In the following example, we use the **length** property of the String object to return the number of characters in a string:

```
<script type="text/javascript">
    var txt="Hello World!";
    document.write(txt.length);
</script>
```

Methods

Methods are the actions that can be performed on objects. Methods are the functions that let the object do something or let something be done to it. There is a small difference between a function and a method, at a function is a standalone unit of statements and a method is attached to an object and can be referenced by the **this** keyword.

Methods are useful for everything from displaying the contents of the object to the screen to performing complex mathematical operations on a group of local properties and parameters.

In the following example, we use the **toUpperCase()** method of the String object

to display a text in uppercase letters:

```
<script type="text/javascript">
    var str="Hello world!";
    document.write(str.toUpperCase());
</script>
```

The output of the previous code will be:

HELLO WORLD!

Programming is all about manipulating data, but what *is* data? ***Data*** is information that we store in our computer programs.

In JavaScript, there are three basic types of data**: numbers, strings, and Booleans**.

JavaScript Number Object

The **Number** object represents numerical date, either integers or floating-point numbers. In general, you do not need to worry about **Number** objects because the browser automatically converts number literals to instances of the number class.

The syntax for creating a **number** object is as follows:

var val = new Number(number);

JavaScript Boolen Object

The **Boolean** object represents two values, either "true" or "false". If *value* parameter is omitted or is 0, -0, null, false, **NaN,** undefined, or the empty string (""), the object has an initial value of false.

Use the following syntax to create a **boolean** object.

var val = new Boolean(value);

JavaScript String Object

The **String** object lets you work with a series of characters; it wraps Javascript's string primitive data type with a number of helper methods. The String object is use to manipulate a stored piece of text. In short, Strings are used to represent text.

As JavaScript automatically converts between string primitives and String objects, you can call any of the helper methods of the String object on a string primitive.

Use the following syntax to create a String object:

var val = new String(string);

JavaScript Arrays Object

The **Array** object lets you store multiple values in a single variable. It stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Use the following syntax to create an **Array** object:

var fruits = new Array("apple", "orange", "mango");

JavaScript Date Object

The Date object is a datatype built into the JavaScript language. Date objects are created with the **new Date( )** as shown below.

Once a Date object is created, a number of methods allow you to operate on it. Most methods simply allow you to get and set the year, month, day, hour, minute, second, and millisecond fields of the object, using either local time or UTC (universal, or GMT) time.

The ECMAScript standard requires the Date object to be able to represent any date and time, to millisecond precision, within 100 million days before or after 1/1/1970. This is a range of plus or minus 273,785 years, so JavaScript can represent date and time till the year 275755.

You can use any of the following syntaxes to create a Date object using Date() constructor.

new Date( )

new Date(milliseconds)

new Date(datestring)

new Date(year,month,date[,hour,minute,second,millisecond])

Note − Parameters in the brackets are always optional.

Here is a description of the parameters −

- **No Argument** − With no arguments, the Date() constructor creates a Date object set to the current date and time.
- **milliseconds** − When one numeric argument is passed, it is taken as the internal numeric representation of the date in milliseconds, as returned by the getTime() method. For example, passing the argument 5000 creates a date that represents five seconds past midnight on 1/1/70.
- **datestring** − When one string argument is passed, it is a string representation of a date, in the format accepted by the **Date.parse()** method.
- **7 agruments** − To use the last form of the constructor shown above. Here is a description of each argument −
  - **year** − Integer value representing the year. For compatibility (in order to avoid the Y2K problem), you should always specify the year in full; use 1998, rather than 98.
  - **month** − Integer value representing the month, beginning with 0 for January to 11 for December.
  - **date** − Integer value representing the day of the month.
  - **hour** − Integer value representing the hour of the day (24-hour scale).
  - **minute** − Integer value representing the minute segment of a time reading.
  - **second** − Integer value representing the second segment of a time reading.

- **millisecond** − Integer value representing the millisecond segment of a time reading.

JavaScript Math Object

The **math** object provides you properties and methods for mathematical constants and functions. Unlike other global objects, **Math** is not a constructor. All the properties and methods of **Math** are static and can be called by using Math as an object without creating it.

Thus, you refer to the constant **pi** as **Math.PI** and you call the *sine* function as **Math.sin(x)**, where x is the method's argument.

The syntax to call the properties and methods of Math are as follows:

```
var pi_val = Math.PI;

var sine_val = Math.sin(30);
```

◀ Preliminary Activity for Week 14 Jump to... Analysis, Application, and Exploration for Week 14 ▶

Navigation

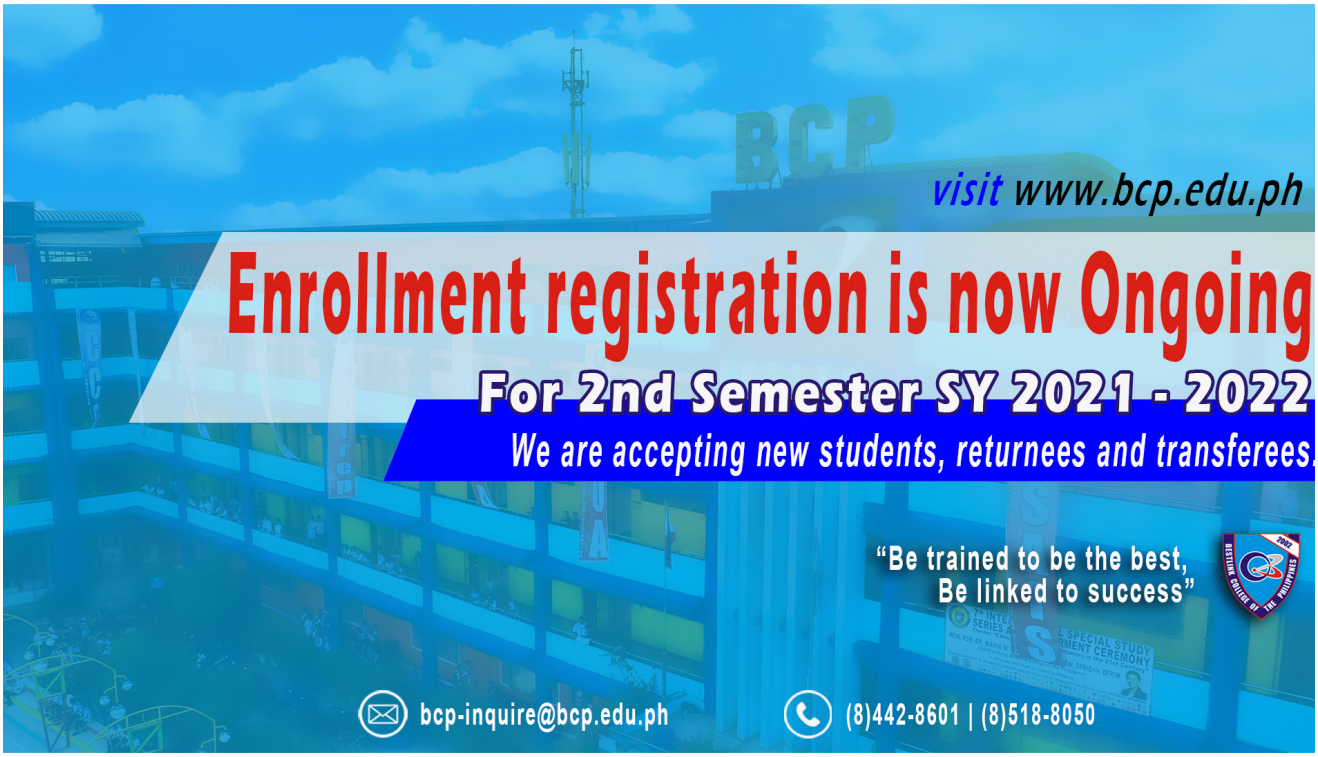
Home
Dashboard
Site pages
My courses
121 - CC106
121 - BPM101 / DM103
121 - OAELEC2
121 - ITE3
121 - MUL101
121 - ITSP2B
121 - WEB101 / CCS3218
Participants
Grades
General
01 Introduction to Internet and World Wide Web
02 Full Stack Development Overview
03 The Basics of HTML

04 Mark Up Text
05 HTML Link, Image and Table
06 - Preliminary Examination
07 HTML Block, Inline Element and Forms Control
08 Introduction To Cascading Style Sheets
09 The Font Style Properties
10 The List and Table style properties
11 Advanced CSS
12 - Midterm Examination
13 JavaScript Basics
14 JavaScript Objects
Preliminary Activity for Week 14
Lesson Proper for Week 14
Analysis, Application, and Exploration for Week 14
Generalization for Week 14
Evaluation for Week 14
Assignment for Week 14
15 Introduction to PHP
16 PHP Control Statement and Function
17 PHP Array and String

Courses

Fair Warning

2nd Semester Enrollment



Activities

- Assignments
- Forums
- Quizzes
- Resources