DepED
DEPARTMENT OF EDUCATION

Technical Vocational Livelihood

11

QUARTER

2

Computer Programming

Computer Programming (ICT) – Grade 11
Quarter 2 – Module 12: JavaScript Objects
First Edition, 2020

Published by the Department of Education Division of Pasig City

Development Team of the Module

Writer : Dan Reinnier C. Amigo
Editor : Ma. Lerma I. Cantanero
Reviewer : Rowena O. Dimagiba
Illustrator:
Layout Artist:
Management Team: Ma. Evalou Concepcion A. Agustin
OIC-Schools Division Superintendent
Dr. Aurelio G. Alfonso
OIC-Assistant Schools Division Superintendent
Dr. Victor Javena
OIC – Chief Curriculum Implementation Division

Education Program Supervisors

Librada L. Agon EdD (EPP/TLE/TVL/TVE)
Liza A. Alvarez (Science/STEM/SSP)
Bernard R. Balitao (AP/HUMSS)
Joselito E. Calios (English/SPFL/GAS)
Norlyn D. Conde EdD (MAPEH/SPA/SPS/HOPE/A&D/Sports)
Wilma Q. Del Rosario (LRMS/ADM)
Ma. Teresita E. Herrera EdD (Filipino/GAS/Piling Larang)
Perlita M. Ignacio PhD (EsP)
Dulce O. Santos PhD (Kindergarten/MTB-MLE)
Teresita P. Tagulao EdD (Mathematics/ABM)

Printed in the Philippines by _____

Department of Education – Division of Pasig City

Office Address: Caruncho Avenue, San Nicolas, Pasig City
Telefax: 641-88-85 / 682-2819
E-mail Address: divisionofpasig@gmail.com

Computer Programming

11

QUARTER 2

MODULE 12

JavaScript Objects

Writer : Dan Reinnier C. Amigo
Editor : Ma. Lerma I. Cantanero
Validator/Reviewer : Rowena O. Dimagiba

Introductory Message

For the Facilitator:

Welcome to the Computer Programming for the ICT Module on JavaScript Objects!

This module was collaboratively designed, developed and reviewed by educators from Schools Division Office of Pasig City headed by its Officer-In-Charge Schools Division Superintendent, Ma. Evalou Concepcion A. Agustin in partnership with the Local Government of Pasig through its mayor, Honorable Victor Ma. Regis N. Sotto. The writers utilized the standards set by the K to 12 Curriculum using the Most Essential Learning Competencies (MELC) while overcoming their personal, social, and economic constraints in schooling.

This learning material hopes to engage the learners into guided and independent learning activities at their own pace and time. Further, this also aims to help learners acquire the needed 21st century skills especially the 5 Cs namely: Communication, Collaboration, Creativity, Critical Thinking and Character while taking into consideration their needs and circumstances.

In addition to the material in the main text, you will also see this box in the body of the module:

Notes to the Teacher

This contains helpful tips or strategies that will help you in guiding the learners.

As a facilitator you are expected to orient the learners on how to use this module. You also need to keep track of the learners' progress while allowing them to manage their own learning. Moreover, you are expected to encourage and assist the learners as they do the tasks included in the module.

For the Learner:

Welcome to the Computer Programming for the ICT Module on JavaScript Objects!

The hand is one of the most symbolized part of the human body. It is often used to depict skill, action and purpose. Through our hands we may learn, create and accomplish. Hence, the hand in this learning resource signifies that you as a learner is capable and empowered to successfully achieve the relevant competencies and skills at your own pace and time. Your academic success lies in your own hands!

This module was designed to provide you with fun and meaningful opportunities for guided and independent learning at your own pace and time. You will be enabled to process the contents of the learning material while being an active learner.

This module has the following parts and corresponding icons:

Expectation - These are what you will be able to know after completing the lessons in the module

Pre-test - This will measure your prior knowledge and the concepts to be mastered throughout the lesson.

Recap - This section will measure what learnings and skills that you understand from the previous lesson.

Lesson- This section will discuss the topic for this module.

Activities - This is a set of activities you will perform.

Wrap Up- This section summarizes the concepts and applications of the lessons.

Valuing-this part will check the integration of values in the learning competency.

Post-test - This will measure how much you have learned from the entire module. Ito po ang parts ng module.

EXPECTATIONS

At the end of this module the learner is expected to:

1. understand JavaScript Objects;
2. differentiate JavaScript objects from functions and variables;
3. execute a code using a JavaScript Object.

PRETEST

Directions: On the space provided before each number, write TRUE if the statement is correct otherwise write FALSE.

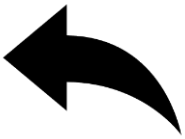
_____ 1. JavaScript objects have properties and methods.

_____ 2. Objects are variables that contain many values.

_____ 3. Spaces and line breaks are important in declaring JavaScript Objects.

_____ 4. A method is a function stored as a property.

_____ 5. Accessing a method without the () parentheses will not result in syntax error.



RECAP

A JavaScript function is a block of code designed to perform a particular task. It is executed when something invokes it. Differentiate JavaScript Function from variable.

LESSON

JavaScript Objects

In real life, a car is an **object**. A car has **properties** like weight and color, and **methods** like start and stop.

Object	Property	Method
Car	car.name = Fiat car.model = 500 car.weight = 850kg car.color = white	car.start() car.drive() car.brake() car.stop()

All cars have the same **properties**, but the property **values** differ from car to car. All cars have the same **methods**, but the methods are performed **at different times.**

Objects are variables that contain many values. Like the example above, many values (Fiat, 500, white) are assigned to the variable named car:

var car = {type: “Fiat”, model: “500”, color: “white”};

The values are written as **name:value** pairs (name and value separated by a colon). JavaScript objects are containers for **named values** called properties or methods.

Object Definition

You define (and create) a JavaScript object with an object literal. Spaces and line breaks are not important. An object definition can span multiple lines:

```
var student = {
        name: “John Smith”,
        age: “19”,
        course: “BSIT”,
        year: “2nd”
};
```

Object Properties

The **name:values** pairs in JavaScript objects are called **properties.**

Property	Property Value
Name	John Smith
Age	19
Course	BSIT
Year	2nd

Accessing Object Properties

Objects are values that can contain other values. They use **keys** to name values, which are a lot like variables.

To get an object's key, you have two options:

1. Using **dot notation** with the name of the key after a period
   - *objectName.propertyName*
2. Using **bracket notation** with the name of the key inside a string inside square brackets [""]
   - *objectName["propertyName"]*

Example 1: dot notation

```
<script>
// Create an object:
var student = {
  name: "John Smith",
  grade : 12,
  course :  "BSIT"
};
// Display some data from the object:
document.getElementById("demo").innerHTML =
student.name + " " + student.grade + student.course;
</script>
```

Example 2: bracket notation

```
<script>
// Create an object:
var student = {
  name: "John Smith",
  grade : 12,
  course :  "BSIT"
};
// Display some data from the object:
document.getElementById("demo").innerHTML =
student["name"] + " " + student["grade"] + student["course"];
</script>
```

Setting Keys

To add keys to an object or overwrite its keys, you have the same two options as getting its keys.

1. Using dot notation with the name of the key after a period and an equals sign =.
2. Using bracket notation with the name of the key inside a string inside square brackets [" "] and an equals sign =.

Example 1: dot notation

```
<script>
var character = {
       name: "Natsu",
       class: "wizard"
};

character.class = "dragon slayer";

character;
document.getElementById("demo").innerHTML =
character.name + " " + character.class;
</script>
```

Example 2: bracket notation

```
<script>
var character = {
       name: "Natsu",
       class: "wizard"
};

character["class"] = "dragon slayer";

character;
document.getElementById("demo").innerHTML =
character.name + " " + character.class;
</script>
```

Object Methods

Objects can also have **methods**. Methods are **actions** that can be performed on objects. Methods are stored in properties as **function definitions**.

Property	Property Value
firstName	Juan
lastName	Dela Cruz
age	19
fullName	function() {return this.firstName + " " + this.lastName;}

Syntax:

```
var person = {
        firstName: "Juan",
        lastName: "Dela Cruz",
        age: 19,
        fullName: function() {
         return this.firstName + " " + this.lastName;
        }
};
```

A method is a function stored as a property.

The "this" Keyword

In a function definition, *this* refers to the "owner" of the function. In the example above, *this* is the **person object** that "owns" the *fullName* function. In other words, *this.firstName* means the *firstName* property of **this object**.

In an object method, *this* refers to the "**owner**" of the method. In the example on the top of this page, *this* refers to the **person** object. The **person** object is the **owner** of the **fullName** method.

```
<p id="demo"></p>

<script>
// Create an object:
var person = {
  firstName: "Juan",
  lastName : "Dela Cruz",
  age      : 19,
  fullName : function() {
    return this.firstName + " " + this.lastName;
  }
};

// Display data from the object:
document.getElementById("demo").innerHTML =
person.fullName();
</script>
```

Accessing Object Methods

You access an object method with the following syntax: objectName.methodName()

Example:

name = person.fullName();

If you access a method **without** the () parentheses, it will return the **function definition.**

Remember: Do Not Declare Strings, Numbers, and Booleans as Objects! When a JavaScript variable is declared with the keyword "**new**", the variable is created as an object:

```
var x = new String();        // Declares x as a String object
var y = new Number();        // Declares y as a Number object
var z = new Boolean();       // Declares z as a Boolean object
```

Avoid **String**, **Number**, and **Boolean** objects. They complicate your code and slow down execution speed.

ACTIVITIES

Creating Syntax: Create an executable syntax by declaring JavaScript Objects. Use the given variables as objects. You may think of your own property and method for each object. Use the rubrics that follows as guide for scoring.

Example:

Given: car

Syntax:

```
Var = car {
        Brand: “Toyota”,
        speed: “120km/h”,
        Stop: function() {
                Return this.brand + “ “ + this.speed;
        }
};
```

1. Smartphone
2. Laptop
3. Cashier

4. Teacher
5. Printer

This rubrics serves as basis for scoring.				
	Needs work	**Developing**	**Meets Standard**	**Score**
Syntax Structure	Syntax is poorly written and exceeds 80 characters per line	Syntax is written with less than 80 characters per line but hard to read	Syntax is well-written	
	5 points	10 points	15 points	
Content	Properties are not relevant to the object. No methods declared.	Properties are relevant to the object. No methods declared.	Properties and methods are relevant to the object.	
	5 points	10 points	15 points	
Code	With syntax and logical errors.	Syntax is correct but with logical errors.	There are no logical and syntax errors.	
	10 points	15 points	20 points	
			Total	**/50**

WRAP–UP

This module discussed the JavaScript Objects, properties and methods. Using a bullet list chart point out how variable, function and object differs from one another.

Variable	Function	Object

VALUING

Directions: Read carefully and answer the following questions.

1. How important it is that you can identify the JavaScript Object from JavaScript Variable? JavaScript Function?

   _____
   _____
   _____
   _____
   _____

2. Why should you consider the execution speed in writing the syntax?

   _____
   _____
   _____

3. What is the relevance of JavaScript objects with real life literal objects?

   _____
   _____
   _____

POST TEST

Directions: Rewrite the underlined word(s) to make each statement correct. If the statement is already correct, write CORRECT. Write your answers on the space before ach number.

1. In a function definition, ***return*** refers to the "owner" of the function.
2. In an object method, ***this*** refers to the "owner" of the object.
3. The syntax for dot notation is: objectName.propertyName
4. The syntax for bracket notation is: bracket notation is: objectName["propertyName"]
5. Avoid String, Number, and Boolean function. They complicate your code and slow down execution speed.

KEY TO CORRECTION

Pre-test:	Post-test:
1. True	1. this
2. True	2. method
3. False	3. CORRECT
4. True	4. CORRECT
5. True	5. objects

Wrap-up

Note: Students can add their own list

Variables

- container for storing data values
- must be identified with unique names
- Names can contain letters, digits, underscores, and dollar signs
- Names are case sensitive

Function

- block of code designed to perform a particular task
- executed when "something" invokes it
- function names can contain letters, digits, underscores, and dollar signs
- Functions stored in variables do not need function names
- Function parameters are listed inside the parentheses ()

Objects

- Objects are variables too
- objects can contain many values
- containers for named values called properties or methods
- Values in objects are called properties
- objects can have methods which are stored in properties as function definitions

References

- "Objects". Accessed September 8, 2020 4:43pm. https://www.javascript.com/learn/objects
- "JavaScript Objects". Accessed September 9, 2020 10:13am. https://www.w3schools.com/js/js_objects.asp
- "The JavaScript this Keyword". Accessed September 9, 2020 11:01am. https://www.w3schools.com/js/js_this.asp