Decision making is a key component of Javascript programming. It allows you to compare values and make decisions upon those values.

We use the Comparison operators to create a condition. The condition MUST be a question that can be answered with a true/yes or a false/no. No other options or answers are valid when a condition is processed. A comparison operator is used between two variables or values. These operators are listed below.

Some conditions may have multiple pieces. These are compound conditions. They use the logical operators below to compare the results of two conditions and produce a valid result. No matter how many pieces the final result of the condition still has to be true/yes or false/no.

**if Statements**

The primary Javascript code used to make comparisons is the if statement. The if statement resolves the associated condition until a final true/yes or false/no result is achieved. Based upon that final value, the if statement will decide which block of code to process or to continue to the next command.

A special form of if statement is called the Switch statement. This structure can replace more complicated if statements in certain cases. The switch statement is very useful when you could have multiple decisions made on the same variable or value. When you hear yourself say something like this: “Based upon the department number we need to do different things” or “If the grade is an A then we print A, if it is a B we need to …”. The switch statement was designed to resolve these types of conditions.

**String Comparisons**

In Javascript compare two strings with the == operator. This is a case sensitive comparison. You can use the toLowerCase( ) or toUpperCase( ) methods of the String object to compare strings without being case sensitive.

**References:**

W3Schools Comparison Operators: <http://www.w3schools.com/js/js_comparisons.asp>

If Statement: <http://www.w3schools.com/js/js_if_else.asp>

Switch Statement: <http://www.w3schools.com/js/js_switch.asp>

Point 1 If Statement syntax

* If statement has several formats depending upon what you need it to do.
* The condition statement of an if statement is always contained within the ( )
* Javascript will resolve the condition down to single result of true/yes or false/no. No matter how complicated the condition statements
* If the condition is true Javascript will process the first block of code after the if statement.
* An if statement must have the true block of code. It may be a single line immediately following the condition or a block of code contained within { }
* If the condition is false Javascript will process the block of code that follows the **else** statement.

***Example:***

**if** ( condition )

{ this code is processed if the condition is true }

**else**

{ this code is process if the condition is false }

Point 2 Equal and Identical Operators

* == equal operator. Use this when you want to compare values or the value of variables within a condition.
* === identical operator. Used when you want to compare values of the value of variables AND they must have the same data type.

***Example:***

var value1 = 5;

var value2 = 5

if ( value1== value2 ) //This condition will return true because the values are the same

if ( value1 === value2) //This conditions will return true because values and data types are the same

Point 3 Logical Operators

* Allows for more complex questions. Usually when you need to say “depending upon”
* && And operator. Both sides of the comparison must be true in order for the condition to be true
* || Or operator. Either side of the comparison may be true and the condition will be true

***Example:***

if ( ( x > 5 ) && (y < 10) ) //both of the inside conditions must be true in order for the whole condition to be true.

if( ( x>5 ) || (y < 10) ) //either of the two inside conditions must be true in order for the condition to be true.

Point 4 One sided if statements

* One sided or unary statement.
* Useful when you only care about one side of the condition. If the condition is true then we need to do something. Otherwise continue.

***Example:***

if ( condition )

{ execute this code if the condition is true }

If the condition is false it will NOT process the code. It will continue processing after the end of the closing }

if ( x > 10 )

{ alert(“You have more than 10 entries”); } //this will process if the condition is true

alert( “Would you like a list of your entries?”); //this will process regardless of the condition

Point 5 Nested If Statement or Handling multiple conditions

* Place an additional if statement within one or both blocks of an if statement
* Used when you need to handle multiple conditions from the same variable or value.

***Example: General format***

if ( condition )

{ process this code if the condition is true

*if ( condition )*

*{ process this code if both conditions are true }*

*else*

*{ process this code if the first condition is true and the second (nested) one is false }*

}//ends the true branch of the first condition

else

{ process this code if the first condition is false }

***Example:***

if ( score < 18 )

{ alert(“You were not successful. Please try again.”); }

else

{

if ( score < 25 )

{ alert(“You were successful. Please continue.”) ; }

else

{ alert( “Good job! Your score is greater than 24.”); }

}

Point 6 Switch Statement

* A more readable method of handling multiple conditions using the same variable/value
* Can be used in place of nested if statements
* Switch only uses the equality or == operator to compare n to the case values.

***Example: General Format***

switch ( n ) //n is a variable or a value

{

case value1: // if ( n == value1 )

code to be executed if n=value1; // process this code if the condition is true

break; //end the switch statement and move to next statement

case value2:

code to be executed n=value2;

break;

default:

code to be executed if none of the cases are valid

}//ends the switch statement

***Example:***

switch (deptNumber)

{

case 100: //Accounting department

alert(“Accounting Department”);

break; //ends switch statement

case 200: //Billing department

alert(“Billing Department”);

break;

default: //if **none** of the other conditions are true

alert(“Department Unavailable”;

}

Point 7 String Comparison

* Use an if statement to compare two string values
* Use the == operator for a case sensitive comparison
* Use the toLowerCase( ) or the toUpperCase( ) methods of the String object for a non case sensitive comparison

var name1 = “Mark”;

var name2 = “MARK”;

***Example:***

if ( name1 == “Mark” ) //Will compare TRUE

{ alert(“The names are the same.”); }

else

{ alert(“The names do not match.”); }

***Example:***

if ( name1 == “MARK” ) //Will compare FALSE

{ alert(“The names are the same.”); }

else

{ alert(“The names do not match.”); }

***Example: Ignores case, both work***

if ( name1.toUpperCase( ) == name2.toUpperCase( ) ) //Puts the value of both variables in all uppercase

{ alert(“The names are the same.”); }

else

{ alert(“The names do not match.”); }

if ( name1.toLowerCase( ) == name2.toLowerCase( ) ) //Puts the value of both variables in all lowercase

{ alert(“The names are the same.”); }

else

{ alert(“The names do not match.”); }