Jared Evans

Professional Instructor
and Lean Process Coach
MasterControl

2018 MASTERS SUMMIT

JavaScript Basics for MasterControl Analytics

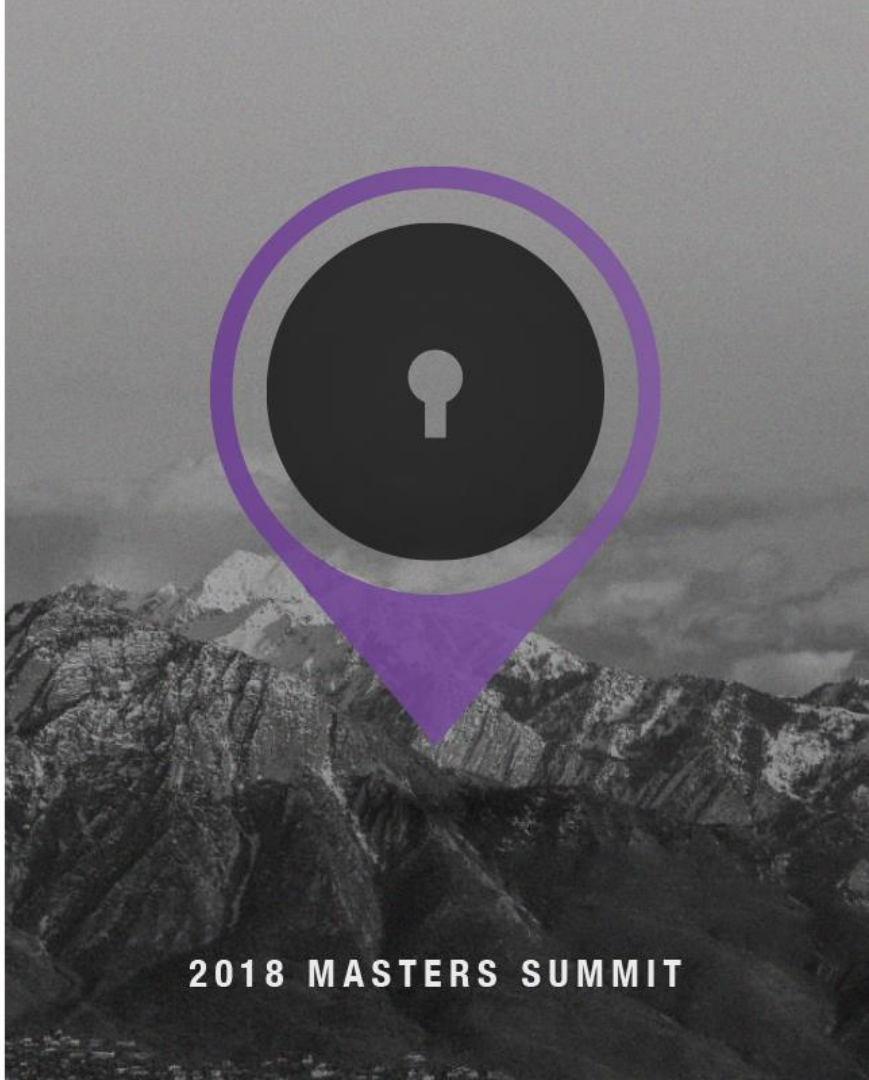
MasterControl

2018 MASTERS SUMMIT

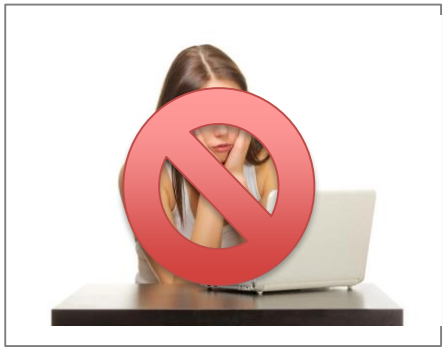
Course Objectives

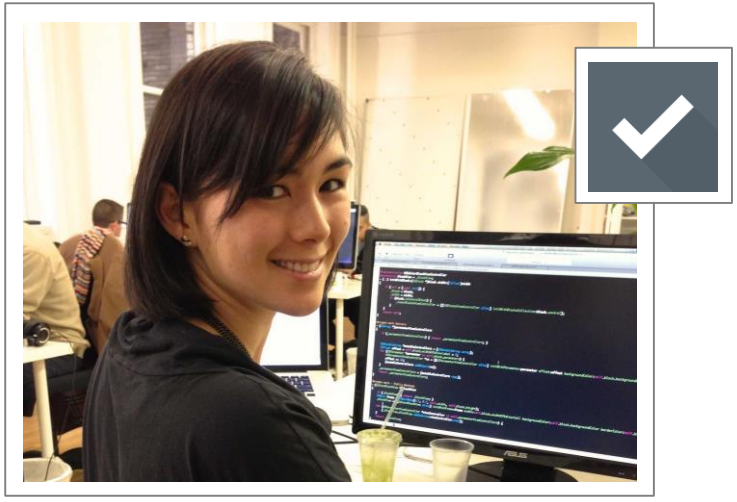
- Become more familiar with JavaScript basics.
- Increase use of the formula editor within MasterControl reports.

MasterControl




MY CODE DOESN'T WORK
I HAVE NO IDEA WHY

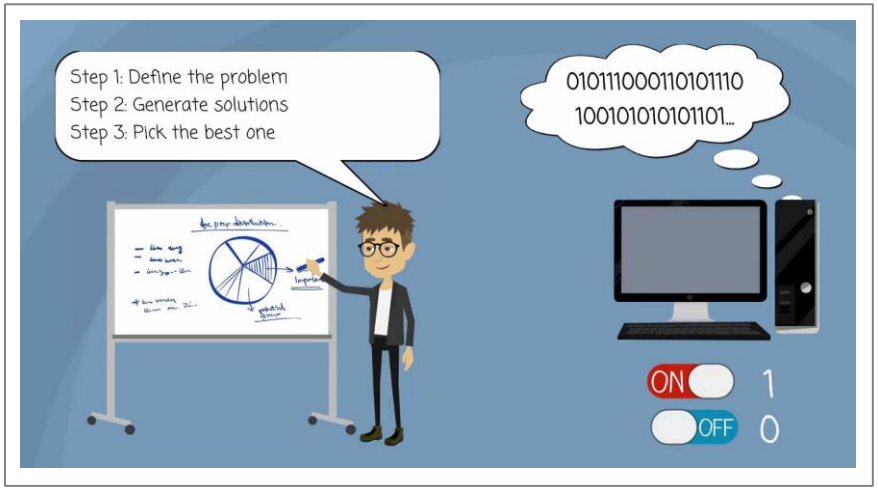


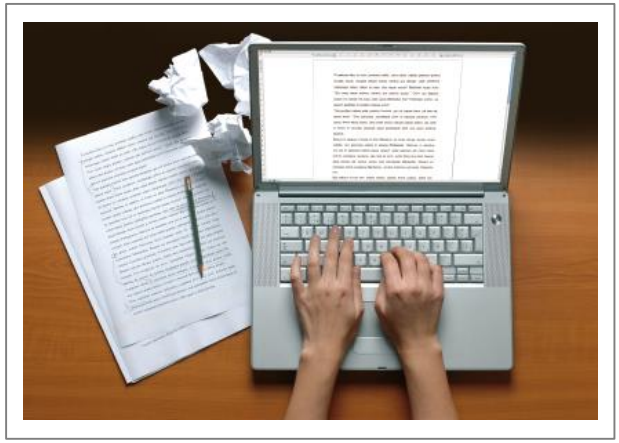


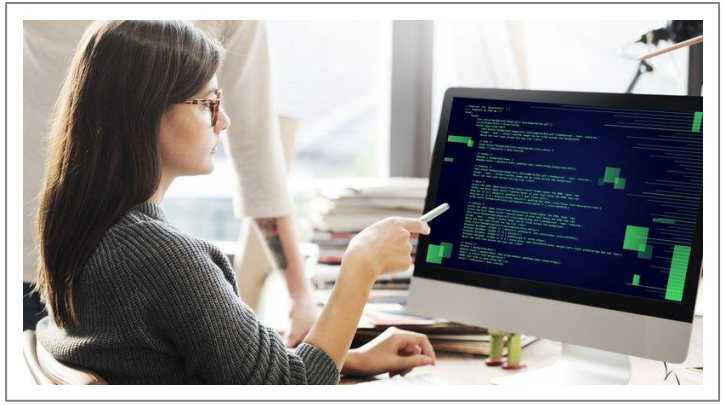
Good Coding Habits

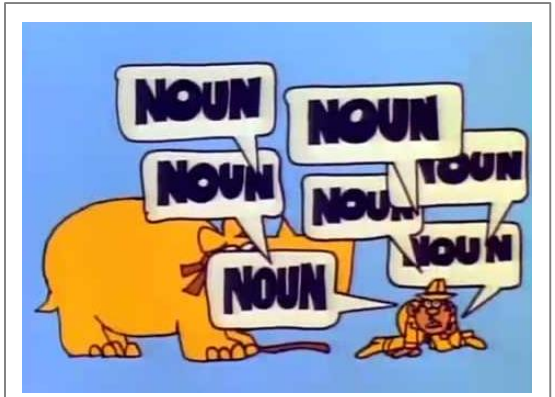
MasterControl

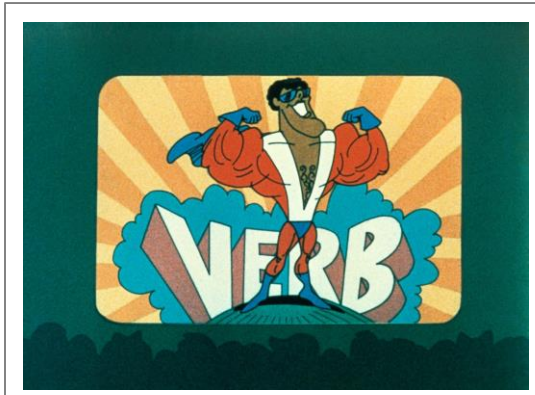
2018 MASTERS SUMMIT


Step 1: Define the problem
Step 2: Generate solutions
Step 3: Pick the best one
0101110001101011 0
100101010101101...
ON
1
OFF
0



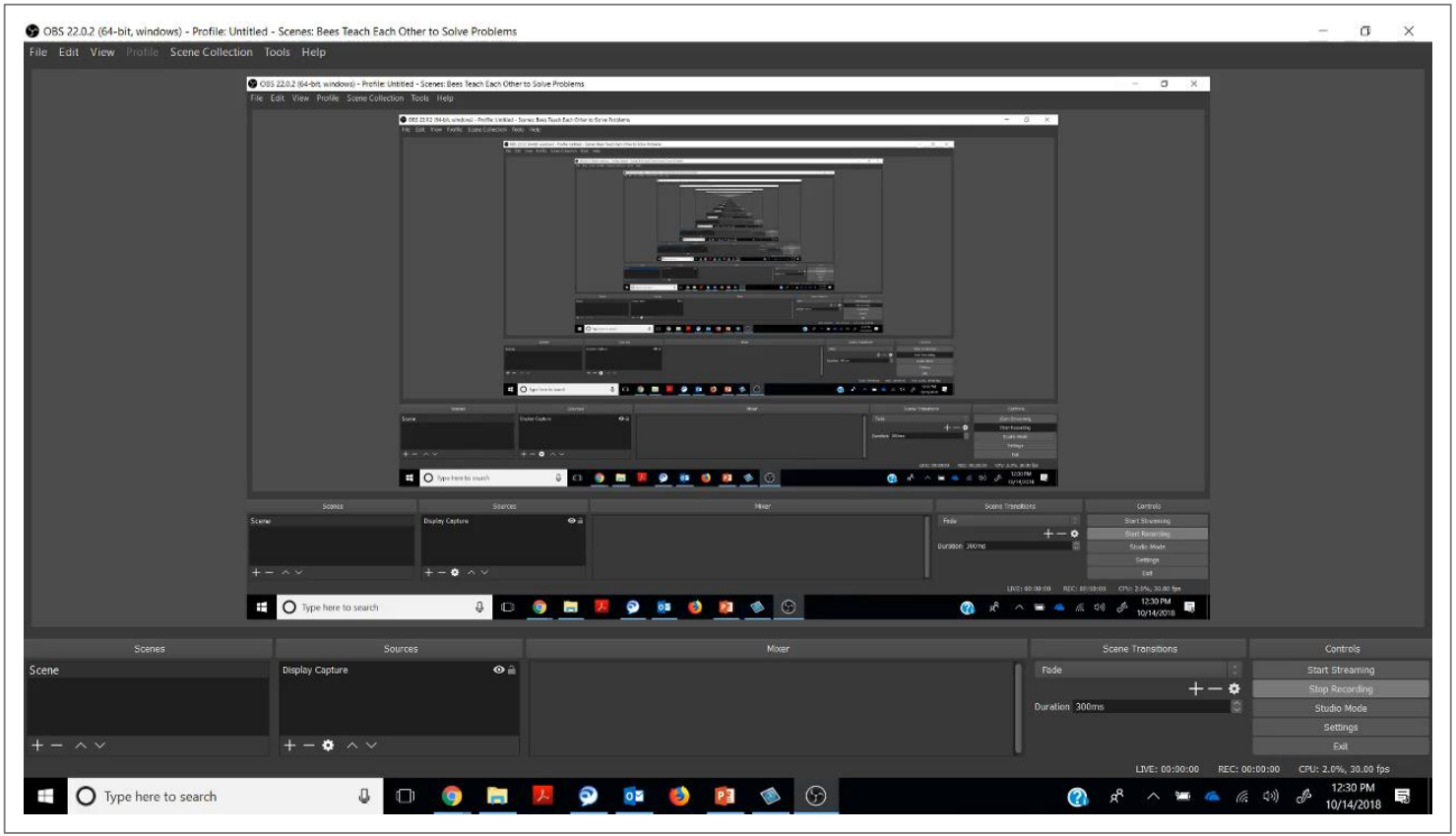


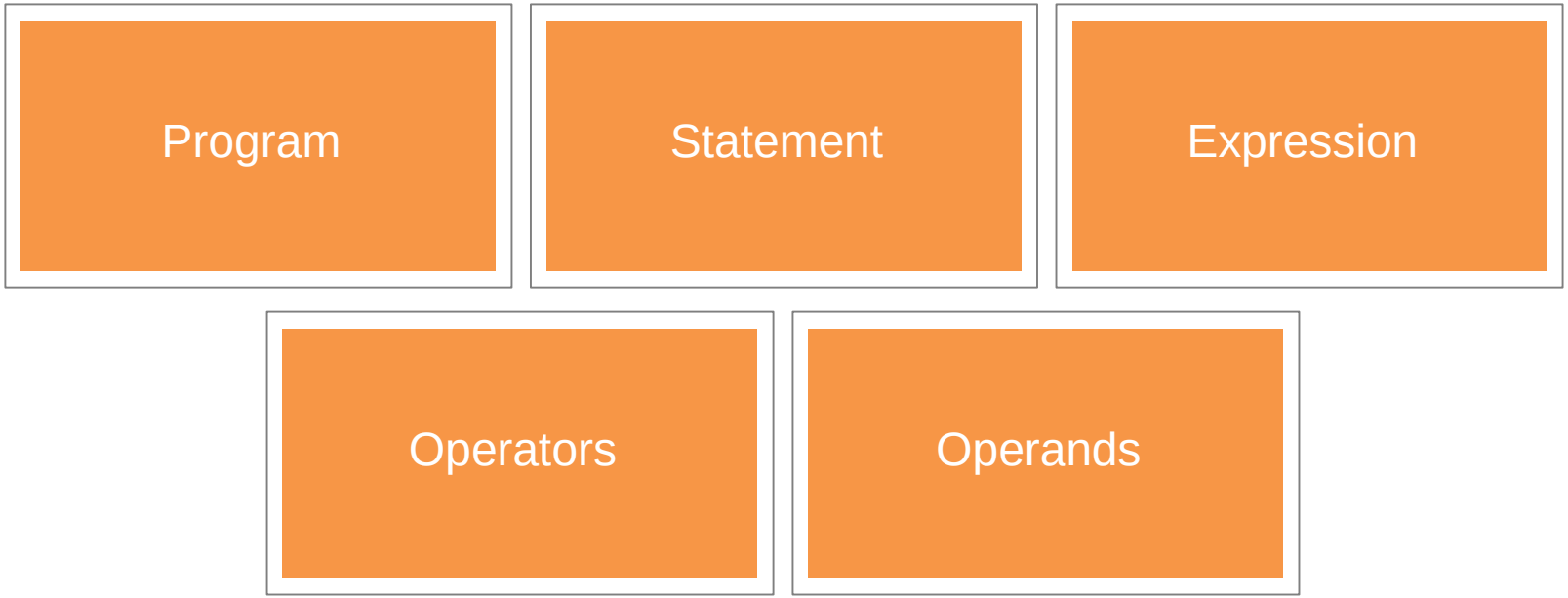

NOUN
NOUN
NOUN
NOUN
NOUN

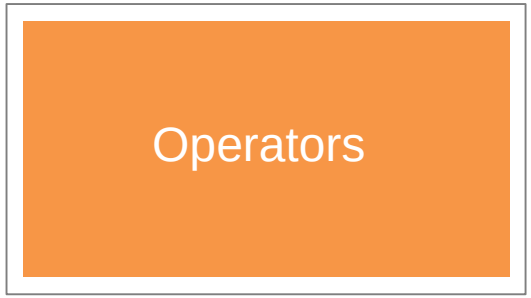

VERB

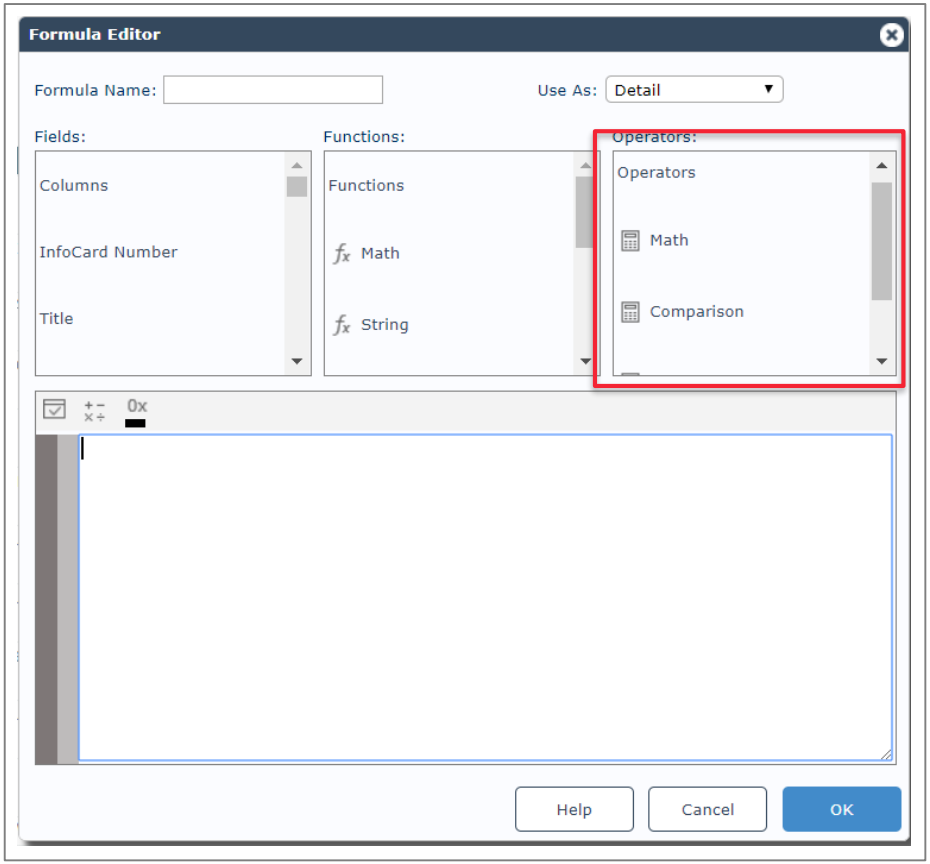

CONJUNCTION JCT.


SCHOOL HOUSE ROCK!


OBS 22.0.2 (64-bit, windows) - Profile: Untitled - Scenes: Bees Teach Each Other to Solve Problems
File Edit View Profile Scene Collection Tools Help
Scenes
Scene
Sources
Display Capture
Mixer
Scene Transitions
Fade
Duration 300ms
Controls
Start Streaming
Stop Recording
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 2.0%, 30.00 fps
Type here to search
12:30 PM
10/14/2018


Program
Statement
Expression
Operators
Operands


Operators


Formula Editor
Formula Name:
Use As:
Detail
Fields:
Functions:
Operators:
Columns
InfoCard Number
Title
Functions
Math
String
Operators
Math
Comparison
0x
Help
Cancel
OK

JavaScript Arithmetic Operators

Arithmetic operators perform arithmetic on numbers (literal or variables).

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus (Remainder)
++	Increment
--	Decrement

JavaScript Assignment Operators

Assignment operators assign values to JavaScript variables.

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
<<=	x <<= y	x = x << y
>>=	x >>= y	x = x >> y
>>>=	x >>>= y	x = x >>> y
&=	x &= y	x = x & y
^=	x ^= y	x = x ^ y
|=	x |= y	x = x | y
**=	x **= y	x = x ** y

Comparison Operators

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

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

Operator	Description	Comparing	Returns
==	equal to	x == 8	false
		x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
		x === "5"	false
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
		x !== 8	true
>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

JavaScript Bitwise Operators

Operator	Name	Description
&	AND	Sets each bit to 1 if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shifts left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shifts right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off
>>>	Zero fill right shift	Shifts right by pushing zeros in from the left, and let the rightmost bits fall off

Conditional (Ternary) Operator

JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.

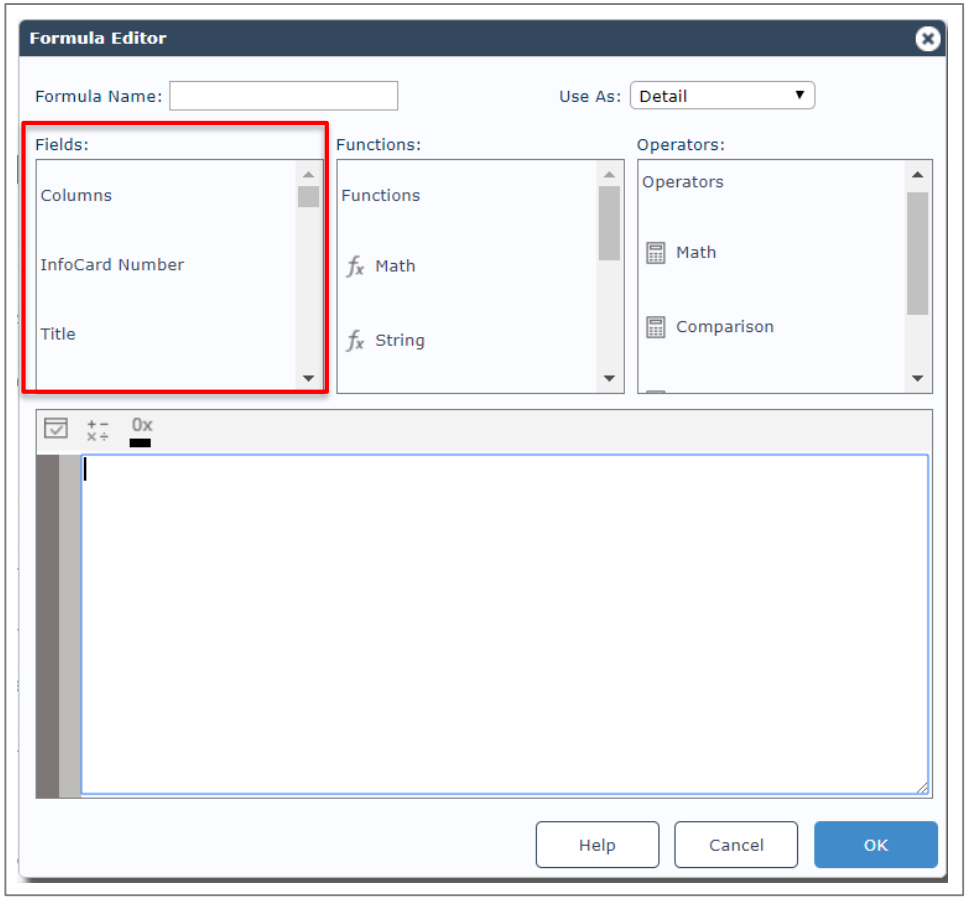
Syntax

```
variablename = (condition) ? value1:value2
```

Example

```
var voteable = (age < 18) ? "Too young":"Old enough";
```

Operands


Formula Editor
Formula Name:
Use As:
Detail
Fields:
Columns
InfoCard Number
Title
Functions:
Functions
Math
String
Operators:
Operators
Math
Comparison
0x
Help
Cancel
OK

Operators and Operands

The numbers (in an arithmetic operation) are called operands.

The operation (to be performed between the two operands) is defined by an operator.

Operand	Operator	Operand
100	+	50


Expression

JavaScript Expressions

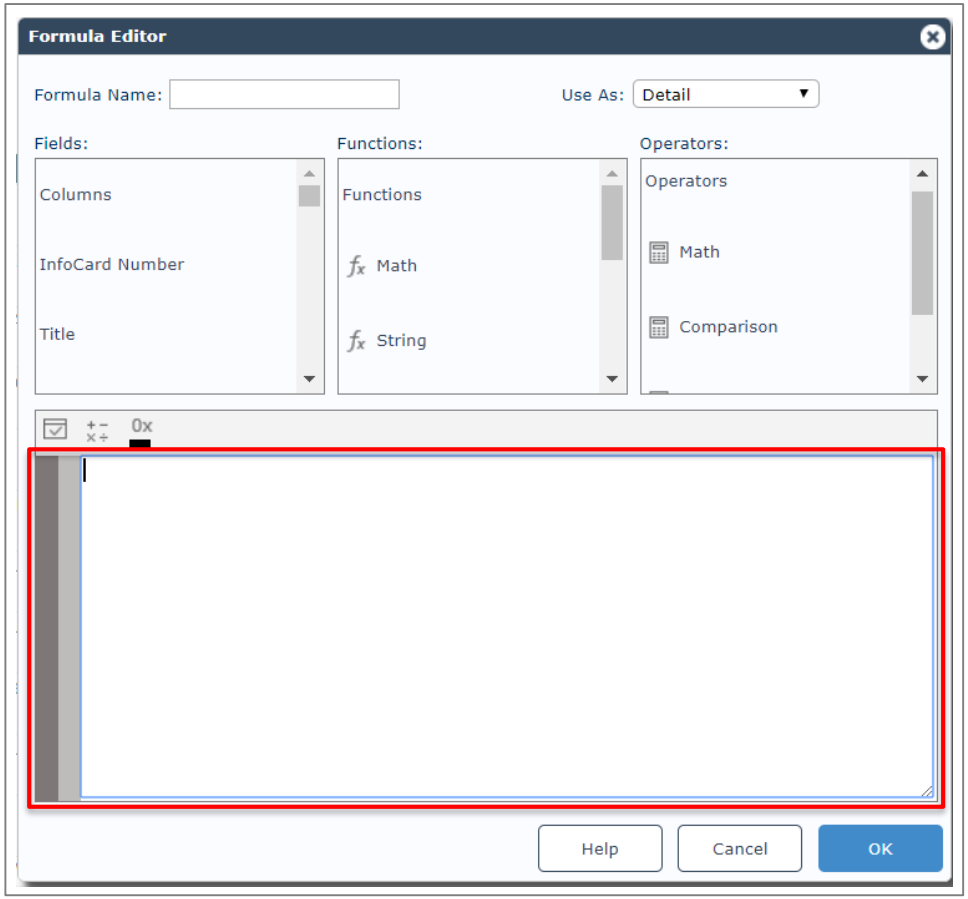
An expression is a combination of values, variables, and operators, which computes to a value.

The computation is called an evaluation.

For example, 5*10 evaluates to 50:

```
5 * 10
```

Statement

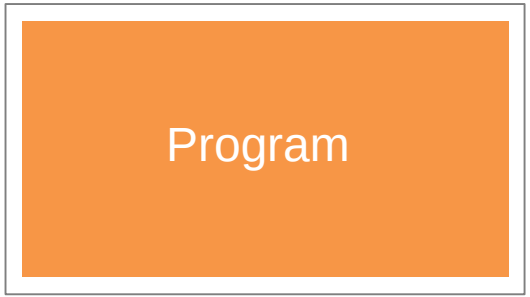

Formula Editor
Formula Name:
Use As:
Detail
Fields:
Functions:
Operators:
Columns
InfoCard Number
Title
Functions
Math
String
Operators
Math
Comparison
0x
Help
Cancel
OK

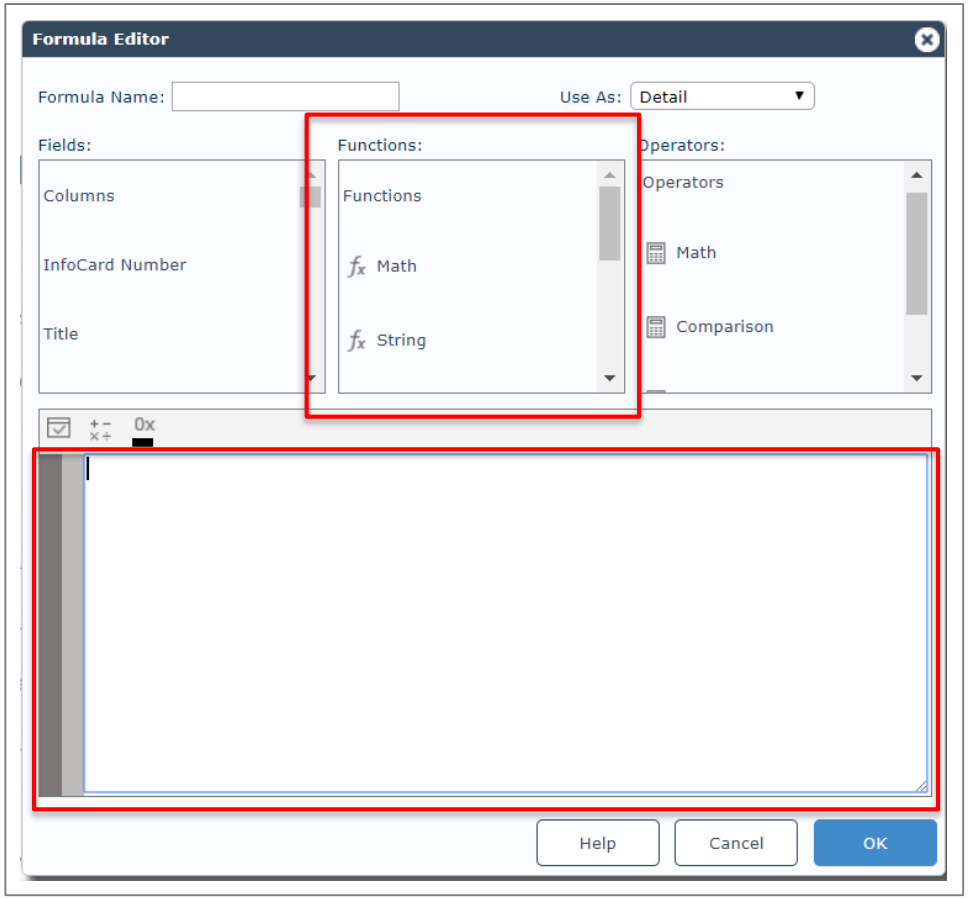
Example

```
var x, y, z;     // Statement 1
x = 5;           // Statement 2
y = 6;           // Statement 3
z = x + y;       // Statement 4
```

Example

```
document.getElementById("demo").innerHTML = "Hello Dolly.";
```


Program


Formula Editor
Formula Name:
Use As:
Detail
Fields:
Functions:
Operators:
Columns
InfoCard Number
Title
Functions
Math
String
Operators
Math
Comparison
0x
Help
Cancel
OK

JavaScript Function Syntax

A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ( ).

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).

The parentheses may include parameter names separated by commas:
(*parameter1, parameter2, ...*)

The code to be executed, by the function, is placed inside curly brackets: **{}**

```
function name(parameter1, parameter2, parameter3) {
    code to be executed
}
```

Function **parameters** are listed inside the parentheses () in the function definition.

Function **arguments** are the **values** received by the function when it is invoked.

Inside the function, the arguments (the parameters) behave as local variables.

```
function myFunction() {
    document.getElementById("demo1").innerHTML = "Hello Dolly!";
    document.getElementById("demo2").innerHTML = "How are you?";
}
```

Example

Calculate the product of two numbers, and return the result:

```
var x = myFunction(4, 3);    // Function is called, return value will end up in x

function myFunction(a, b) {
    return a * b;            // Function returns the product of a and b
}
```

The result in x will be:

```
12
```

Convert Fahrenheit to Celsius:

```
function toCelsius(fahrenheit) {
    return (5/9) * (fahrenheit-32);
}
document.getElementById("demo").innerHTML = toCelsius(77);
```

Single-Line Comments

Single-line comments start with //.

Any text between // and the end of the line will be ignored by JavaScript (will not be executed).

Multi-Line Comments

Multi-line comments start with /* and end with */.

Any text between /* and */ will be ignored by JavaScript.

```
var x = 5;   // I will be executed

// var x = 6;   I will NOT be executed
```

JavaScript Is Case Sensitive

All JavaScript **identifiers** are **case sensitive**.

The variables lastName and lastname are two different variables.

```
var lastname, lastName;
lastName = "Doe";
lastname = "Peterson";
```

Variable/Data Types

A place where info can be stored and retrieved.
Identifies the kind of data being stored.

- Number = Math, algebra, +, -, or decimal.
- String = A string of characters seen on screen “ “.
- Object = A noun including properties and methods.
- Boolean = Yes/No or True/False.
- Array = A variable that can hold many variables.
- Undefined = Variable with no value assigned to it.
- Null = Represents no value.

Naming Rules for Identifiers/variables

1. Begin with either a letter, dollar sign or underscore.
2. No other special characters.
3. No spaces between two words which would be one variable/name/identifier.
4. Can't use JavaScript keywords.
5. They are case sensitive.
6. Should be descriptive.
7. When using multiple words, use camel casing (first letter of first word is lower case, any additional words in the name begin with an uppercase with no space before them).
8. Be consistent by following the same naming convention.

JavaScript Keywords

JavaScript statements often start with a keyword to identify the JavaScript action to be performed.

Keyword	Description
break	Terminates a switch or a loop
continue	Jumps out of a loop and starts at the top
debugger	Stops the execution of JavaScript, and calls (if available) the debugging function
do ... while	Executes a block of statements, and repeats the block, while a condition is true
for	Marks a block of statements to be executed, as long as a condition is true
function	Declares a function
if ... else	Marks a block of statements to be executed, depending on a condition
return	Exits a function
switch	Marks a block of statements to be executed, depending on different cases
try ... catch	Implements error handling to a block of statements
var	Declares a variable

JavaScript Statements

MasterControl

2018 MASTERS SUMMIT

Conditional Statements

Very often when you write code, you want to perform different actions for different decisions.

You can use conditional statements in your code to do this.

In JavaScript we have the following conditional statements:

- Use **if** to specify a block of code to be executed, if a specified condition is true
- Use **else** to specify a block of code to be executed, if the same condition is false
- Use **else if** to specify a new condition to test, if the first condition is false
- Use **switch** to specify many alternative blocks of code to be executed

The “if” Statement

Use the "if" statement to specify a block of JavaScript code to be executed if a condition is true.

Syntax

```
if (condition) {
    block of code to be executed if the condition is true
}
```

Note that **if** is in lowercase letters. Uppercase letters (If or IF) will generate a JavaScript error.

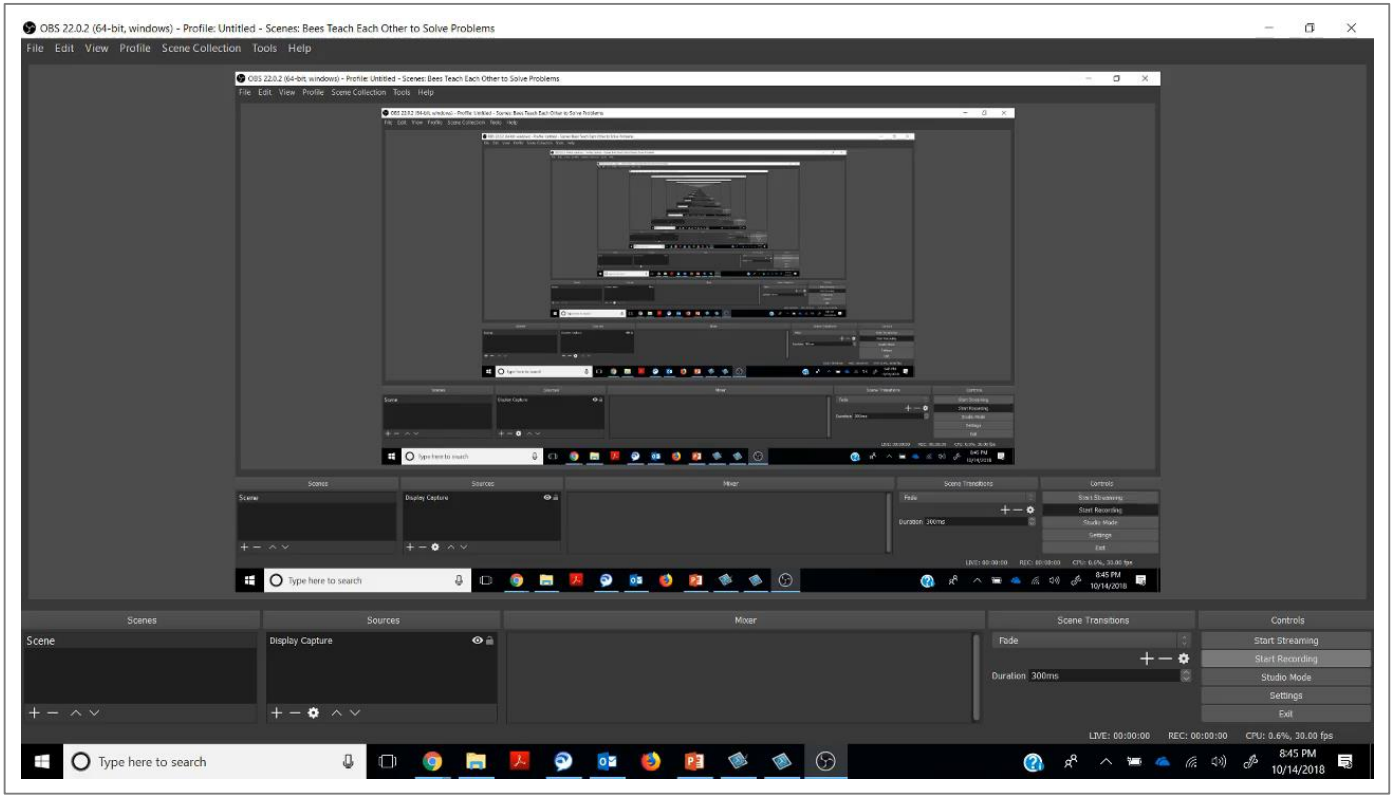
Example

Make a "Good day" greeting if the hour is less than 18:00:

```
if (hour < 18) {
    greeting = "Good day";
}
```

The result of greeting will be:

```
Good day
```


OBS 22.0.2 (64-bit, windows) - Profile: Untitled - Scenes: Bees Teach Each Other to Solve Problems
File Edit View Profile Scene Collection Tools Help
Scenes
Scene
Sources
Display Capture
Mixer
Scene Transitions
Fade
Duration 300ms
Controls
Start Streaming
Start Recording
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 0.6%, 30.00 fps
Type here to search
8:45 PM
10/14/2018

The “else” Statement

Use the "else" statement to specify a block of code to be executed if the condition is false.

```
if (condition) {
    block of code to be executed if the condition is true
} else {
    block of code to be executed if the condition is false
}
```

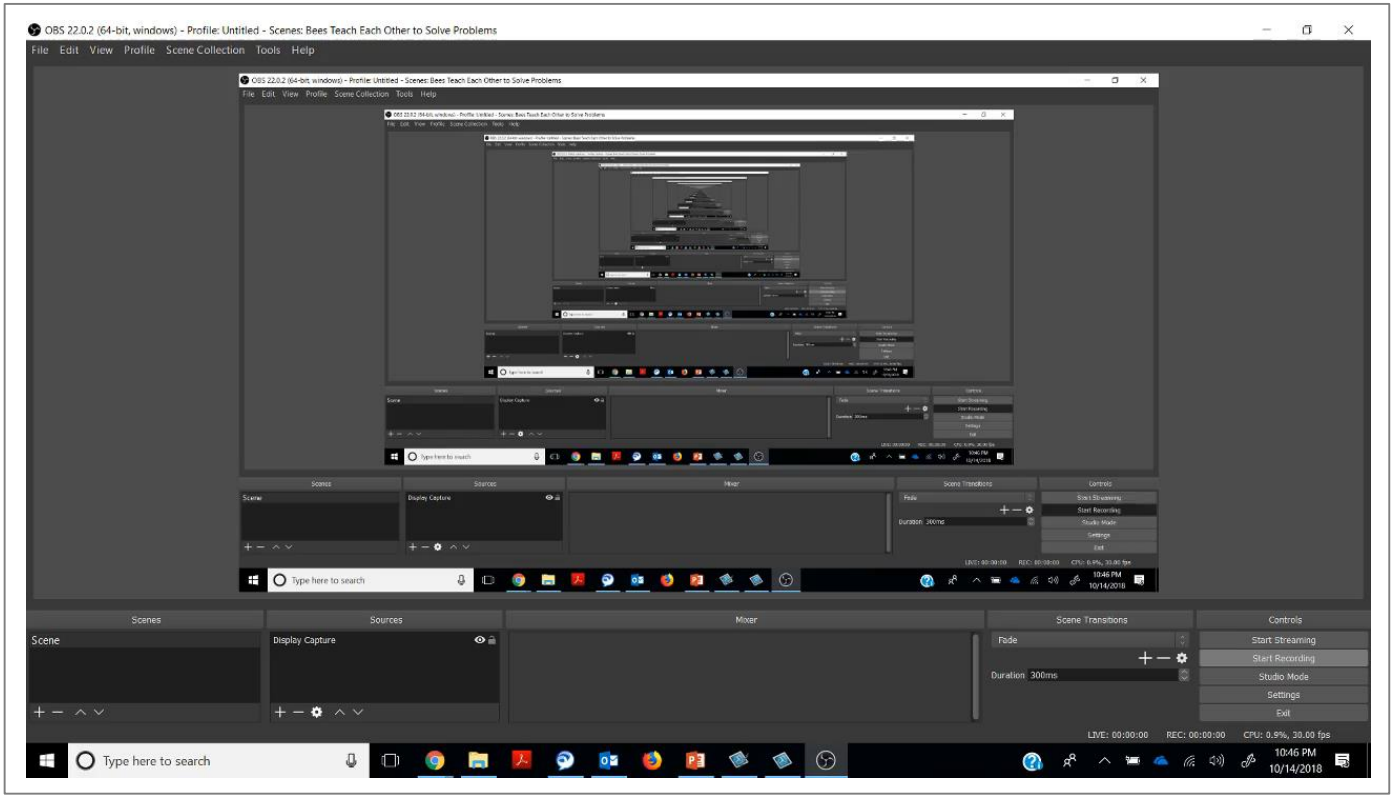
Example

If the hour is less than 18, create a "Good day" greeting, otherwise "Good evening":

```
if (hour < 18) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

The result of greeting will be:

```
Good day
```


OBS 22.0.2 (64-bit, windows) - Profile: Untitled - Scenes: Bees Teach Each Other to Solve Problems
File Edit View Profile Scene Collection Tools Help
Scenes
Scene
Sources
Display Capture
Mixer
Scene Transitions
Fade
Duration 300ms
Controls
Start Streaming
Start Recording
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 0.9%, 30.00 fps
Type here to search
10:46 PM
10/14/2018

The “else if” Statement

Use the "else if" statement to specify a new condition if the first condition is false.

Syntax

```
if (condition1) {
    block of code to be executed if condition1 is true
} else if (condition2) {
    block of code to be executed if the condition1 is false and condition2 is true
} else {
    block of code to be executed if the condition1 is false and condition2 is false
}
```

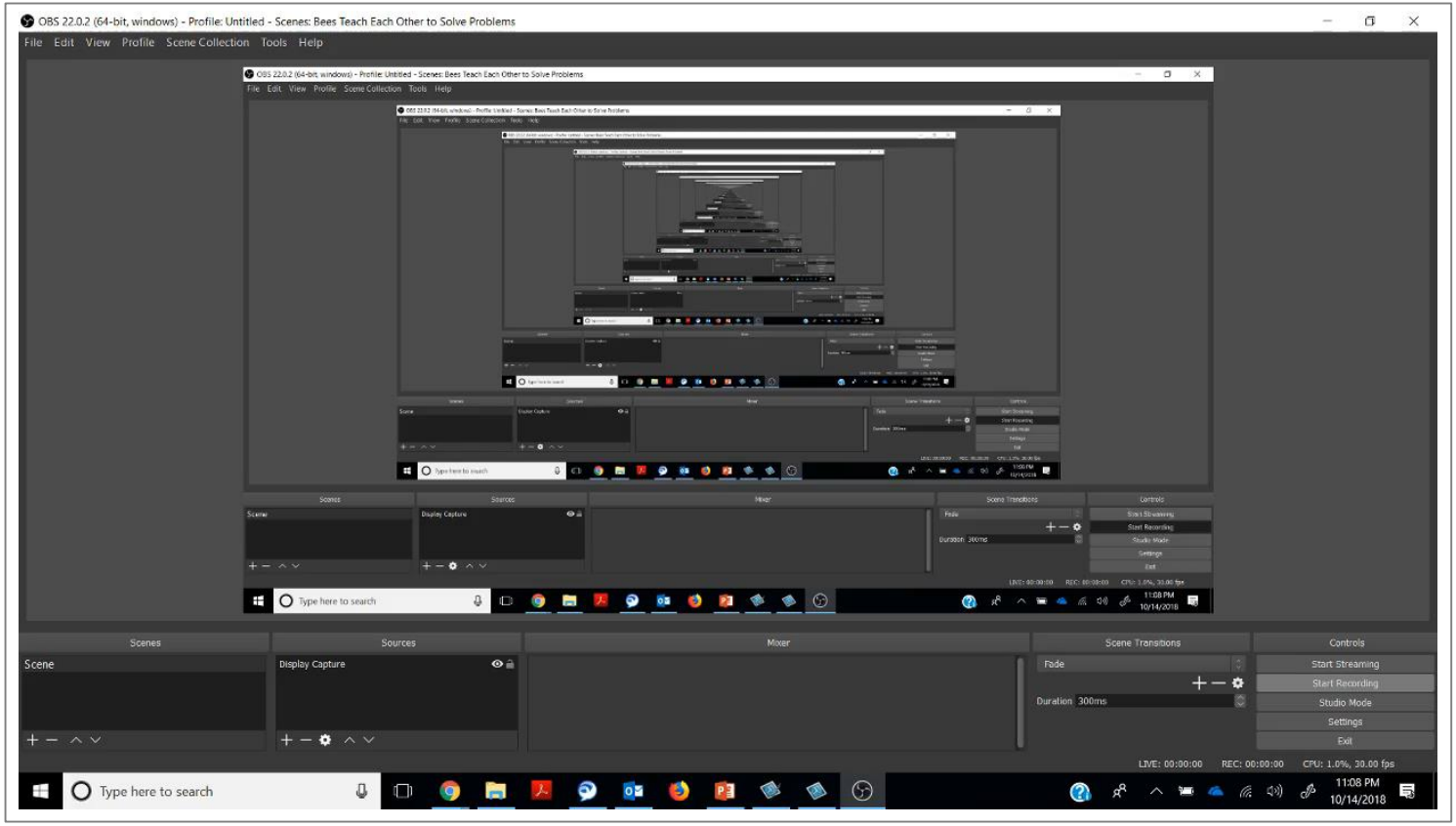
Example

If time is less than 10:00, create a "Good morning" greeting, if not, but time is less than 20:00, create a "Good day" greeting, otherwise a "Good evening":

```
if (time < 10) {
    greeting = "Good morning";
} else if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

The result of greeting will be:

```
Good day
```


OBS 22.0.2 (64-bit, windows) - Profile: Untitled - Scenes: Bees Teach Each Other to Solve Problems
File Edit View Profile Scene Collection Tools Help
Scenes
Scene
Sources
Display Capture
Mixer
Scene Transitions
Fade
Duration 300ms
Controls
Start Streaming
Start Recording
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 1.0%, 30.00 fps
Type here to search
11:08 PM
10/14/2018

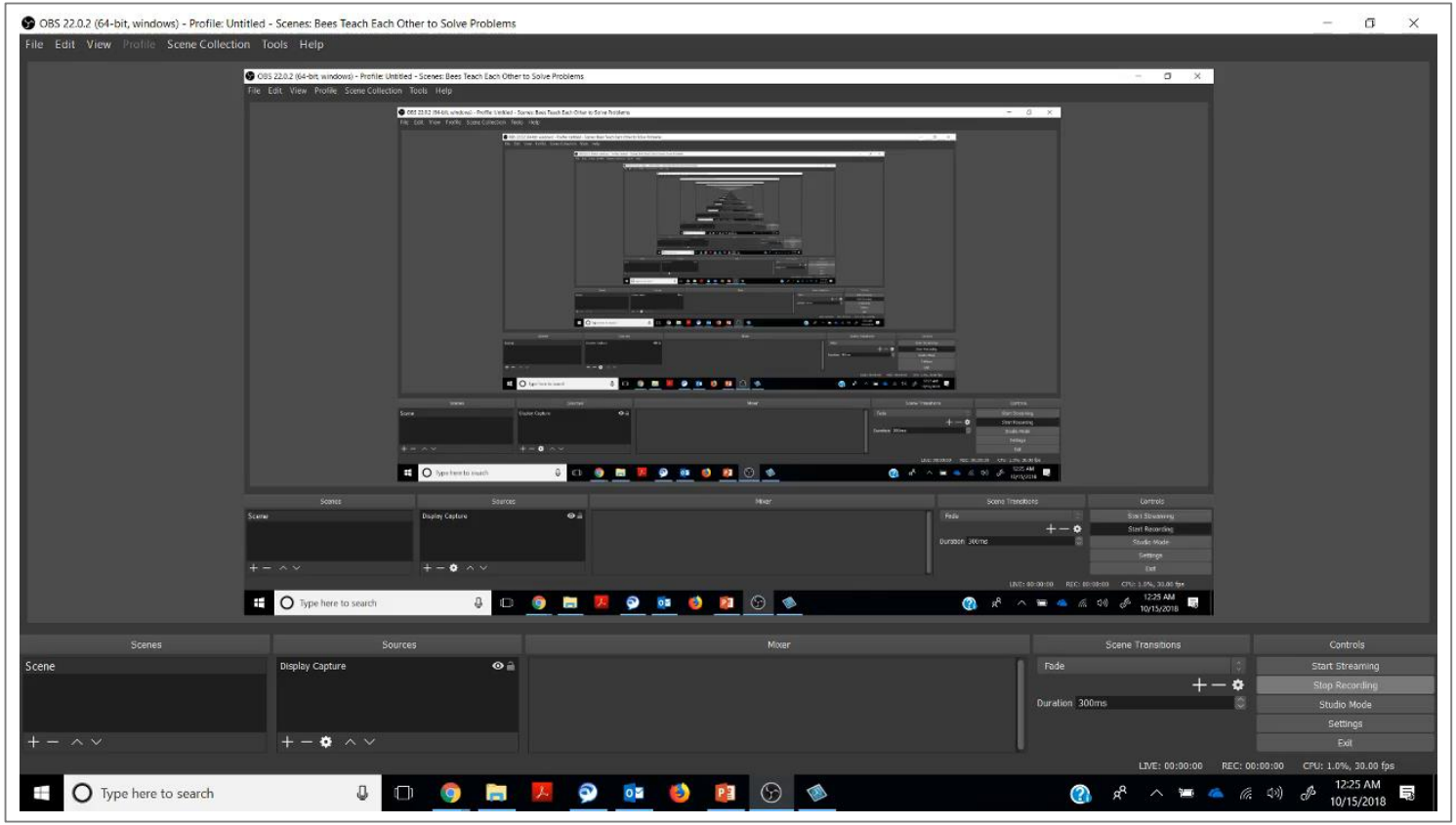
The JavaScript Switch Statement

Use the switch statement to select one of many code blocks to be executed.

- This is how it works:
- The switch expression is evaluated once.
- The value of the expression is compared with the values of each case.
- If there is a match, the associated block of code is executed.

Syntax

```
switch(expression) {
    case x:
        code block
        break;
    case y:
        code block
        break;
    default:
        code block
}
```


OBS 22.0.2 (64-bit, windows) - Profile: Untitled - Scenes: Bees Teach Each Other to Solve Problems
File Edit View Profile Scene Collection Tools Help
Scenes
Scene
Sources
Display Capture
Mixer
Scene Transitions
Fade
Duration 300ms
Controls
Start Streaming
Stop Recording
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 1.0%, 30.00 fps
Type here to search
12:25 AM
10/15/2018

JavaScript Loops

Loops can execute a block of code a number of times.

Loops are handy, if you want to run the same code over and over again, each time with a different value.

Often this is the case when working with arrays:

Instead of writing:

```
text += cars[0] + "<br>";
text += cars[1] + "<br>";
text += cars[2] + "<br>";
text += cars[3] + "<br>";
text += cars[4] + "<br>";
text += cars[5] + "<br>";
```

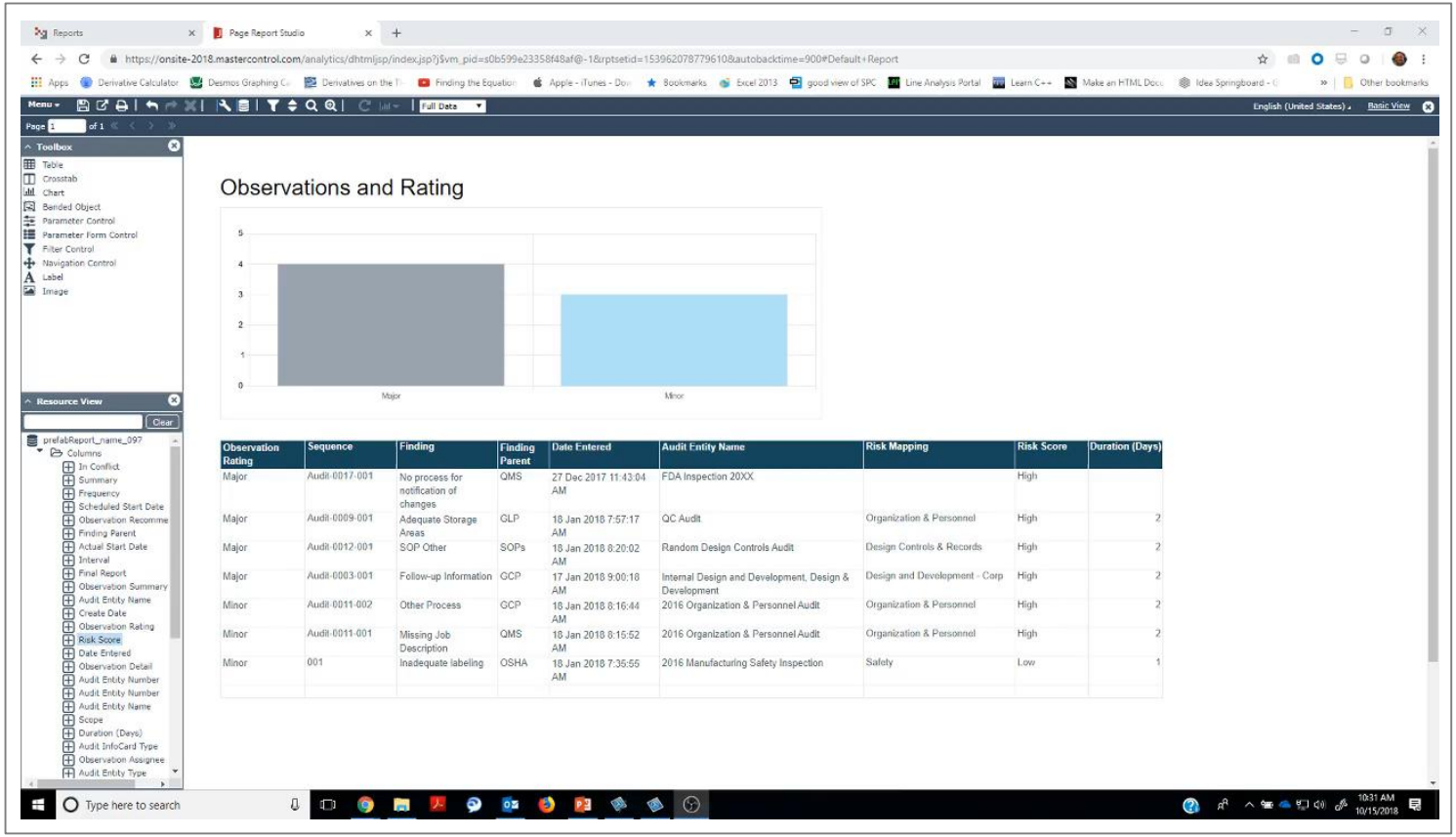
You can write:

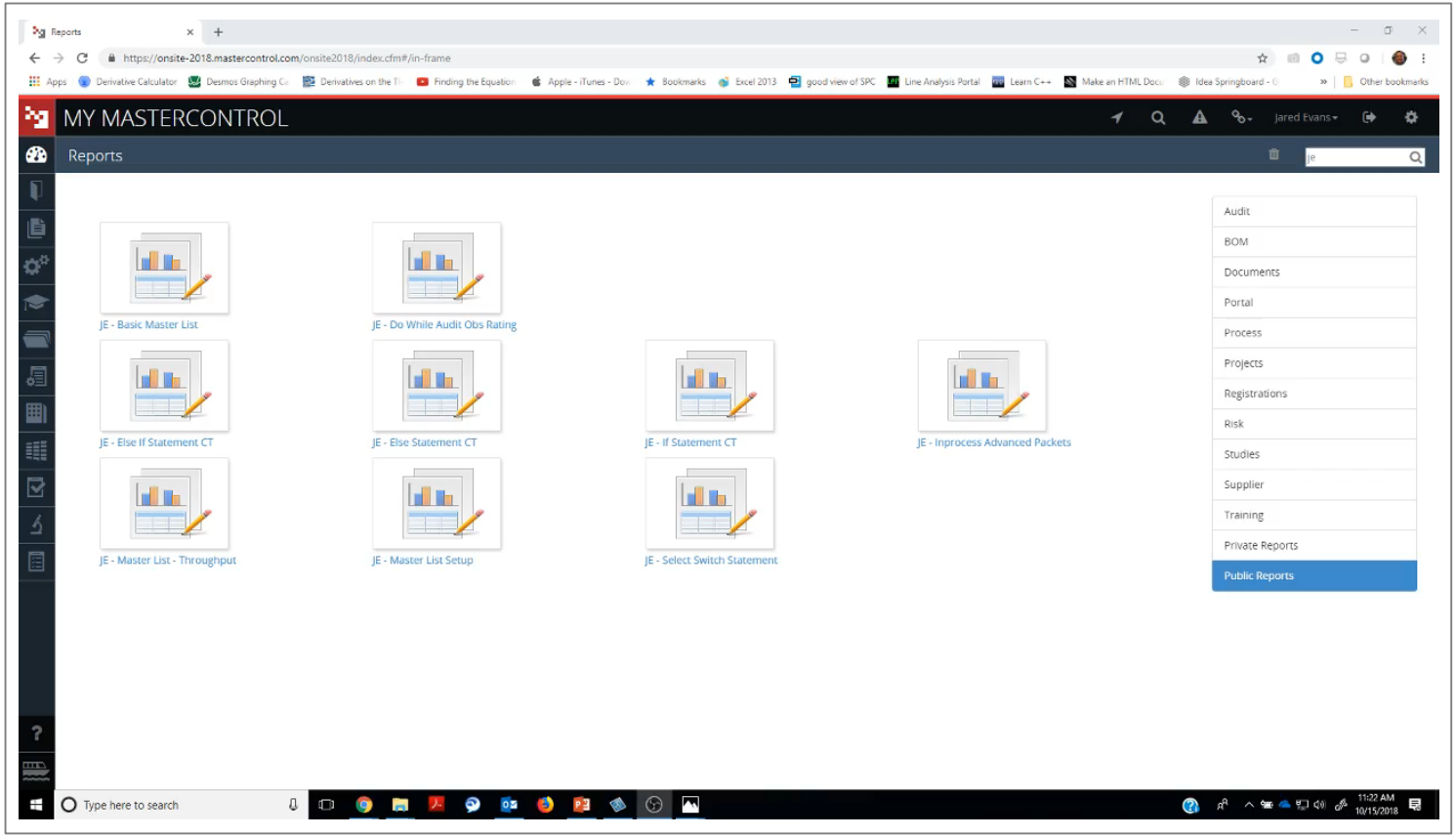
```
var i;
for (i = 0; i < cars.length; i++) {
    text += cars[i] + "<br>";
}
```

Different Kinds of Loops

JavaScript supports different kinds of loops:

- **for** - loops through a block of code a number of times
- **for/in** - loops through the properties of an object
- **while** - loops through a block of code while a specified condition is true
- **do/while** - also loops through a block of code while a specified condition is true

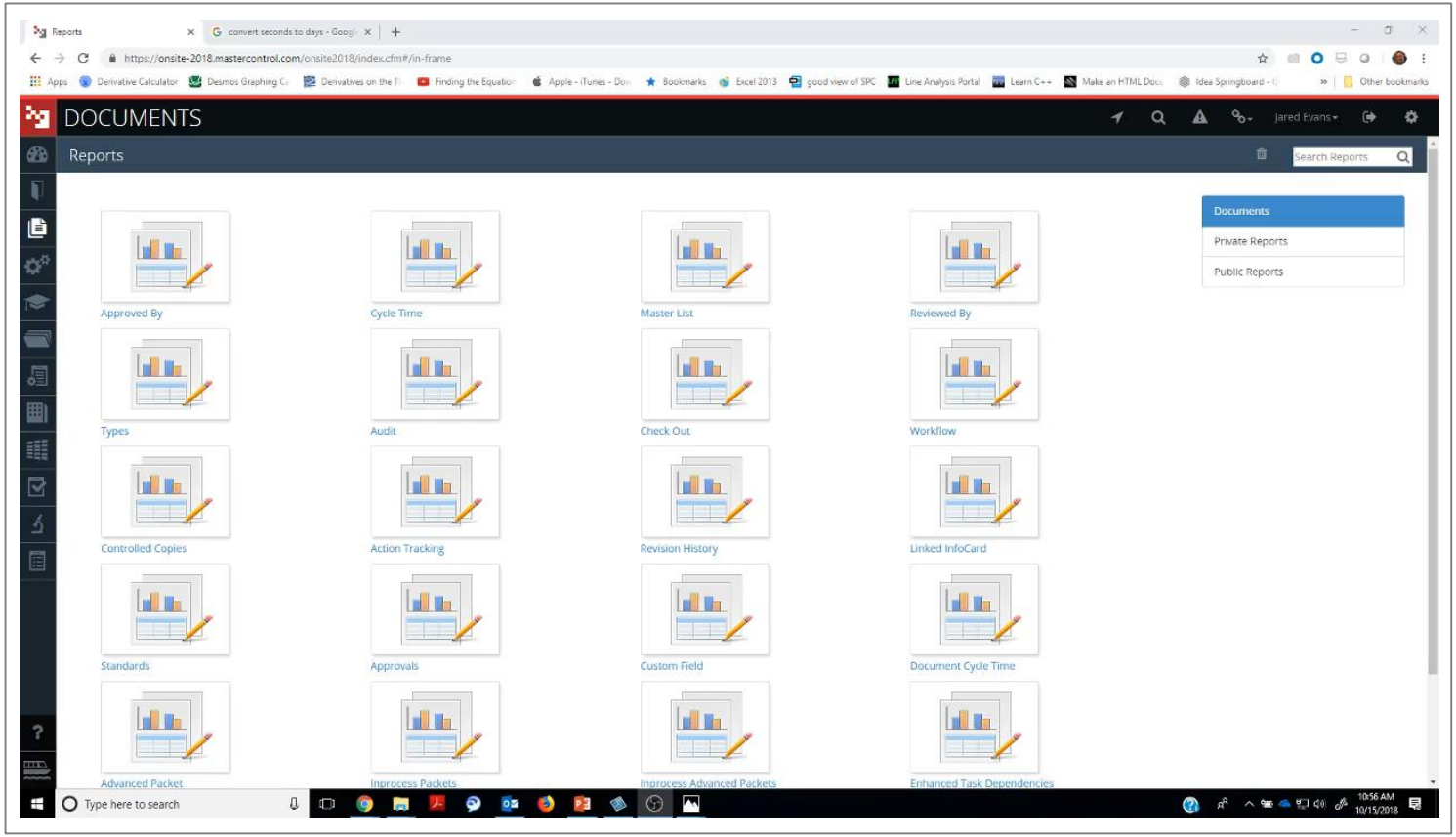

Reports
Page Report Studio
https://onsite-2018.mastercontrol.com/analytics/dhtmljsp/index.jsp?jvm_pid=s0b599e23358f48af@-1&rptsetid=1539620787796108autobacktime=900#Default+Report
Toolbox
Table
Crosstab
Chart
Banded Object
Parameter Control
Parameter Form Control
Filter Control
Navigation Control
Label
Image
Resource View
Observations and Rating
Major
Minor
Observation Rating
Sequence
Finding
Finding Parent
Date Entered
Audit Entity Name
Risk Mapping
Risk Score
Duration (Days)
Major
Audit-0017-001
No process for notification of changes
QMS
27 Dec 2017 11:43:04 AM
FDA Inspection 20XX
High
Major
Audit-0009-001
Adequate Storage Areas
GLP
18 Jan 2018 7:57:17 AM
QC Audit
Organization & Personnel
High
2
Major
Audit-0012-001
SOP Other
SOPs
18 Jan 2018 8:20:02 AM
Random Design Controls Audit
Design Controls & Records
High
2
Major
Audit-0003-001
Follow-up Information
GCP
17 Jan 2018 9:00:18 AM
Internal Design and Development, Design & Development
Design and Development - Corp
High
2
Minor
Audit-0011-002
Other Process
GCP
18 Jan 2018 8:16:44 AM
2016 Organization & Personnel Audit
Organization & Personnel
High
2
Minor
Audit-0011-001
Missing Job Description
QMS
18 Jan 2018 8:15:52 AM
2016 Organization & Personnel Audit
Organization & Personnel
High
2
Minor
001
Inadequate labeling
OSHA
18 Jan 2018 7:35:55 AM
2016 Manufacturing Safety Inspection
Safety
Low
1
Type here to search

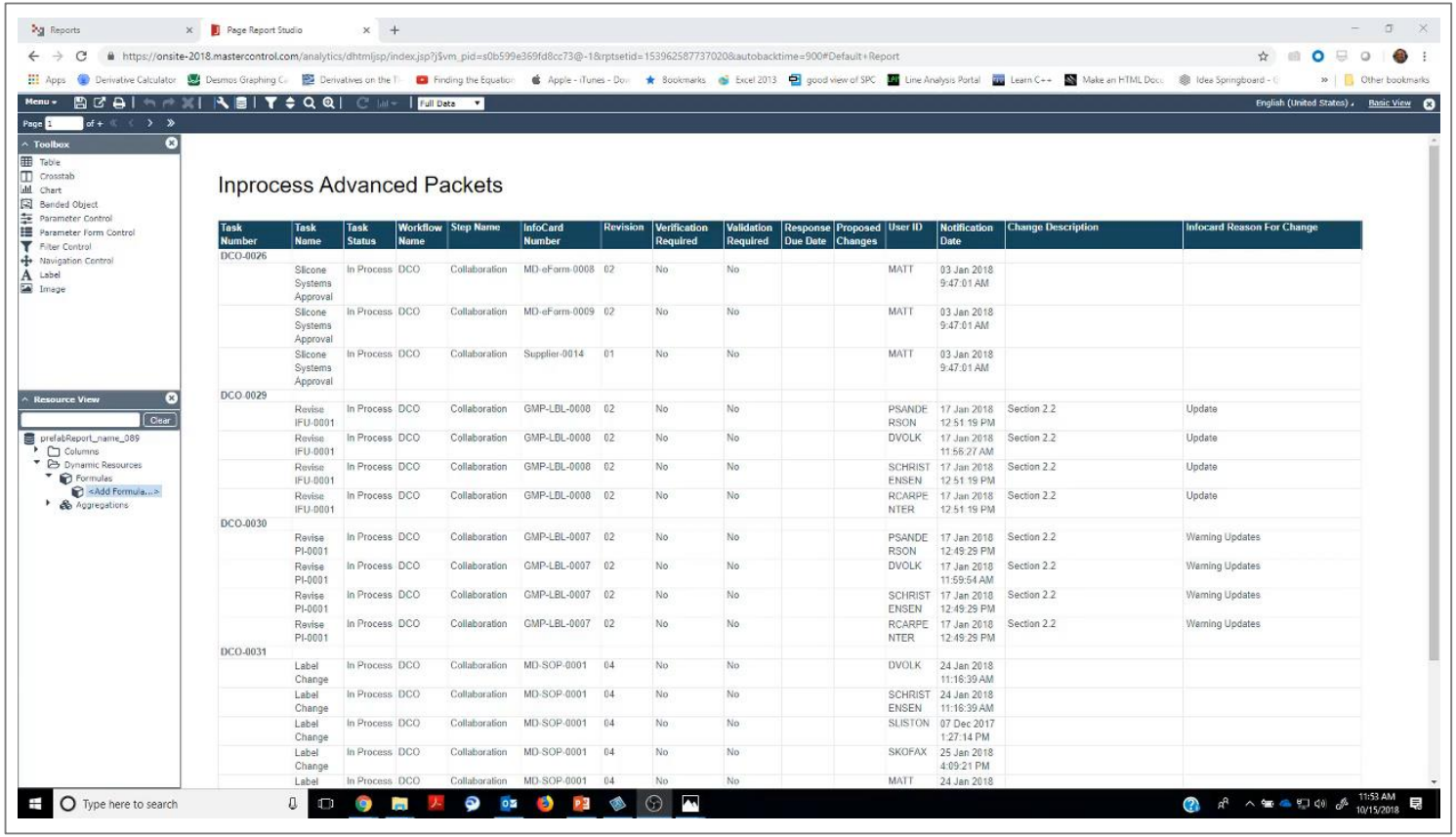

Reports
https://onsite-2018.mastercontrol.com/onsite2018/index.cfm#/in-frame
MY MASTERCONTROL
Jared Evans
Reports
JE - Basic Master List
JE - Do While Audit Obs Rating
JE - Else If Statement CT
JE - Else Statement CT
JE - If Statement CT
JE - Inprocess Advanced Packets
JE - Master List - Throughput
JE - Master List Setup
JE - Select Switch Statement
Audit
BOM
Documents
Portal
Process
Projects
Registrations
Risk
Studies
Supplier
Training
Private Reports
Public Reports
Type here to search
11:22 AM
10/15/2018

Additional Formula Examples

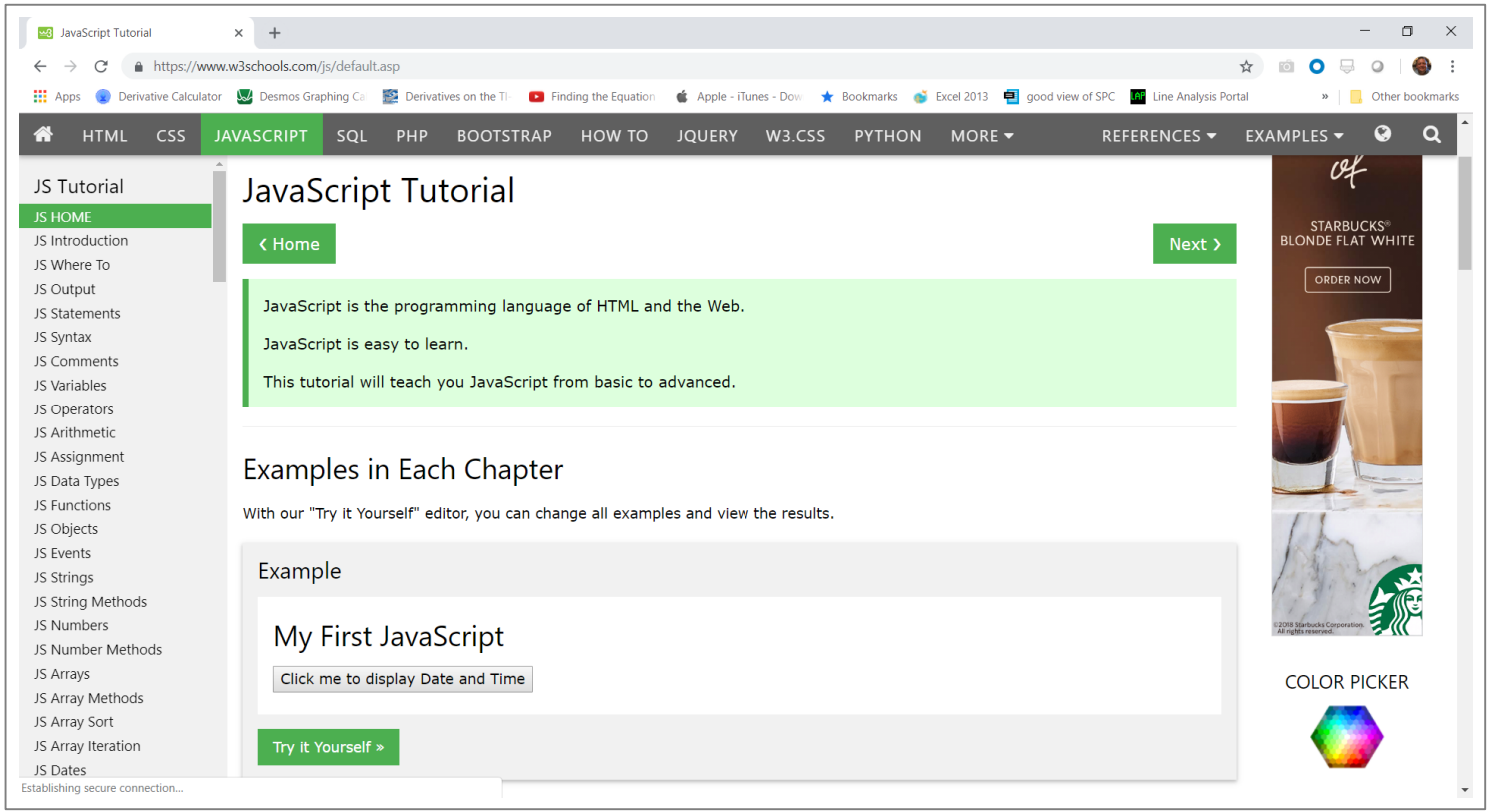
MasterControl

2018 MASTERS SUMMIT


DOCUMENTS
Reports
Search Reports
Documents
Private Reports
Public Reports
Approved By
Cycle Time
Master List
Reviewed By
Types
Audit
Check Out
Workflow
Controlled Copies
Action Tracking
Revision History
Linked InfoCard
Standards
Approvals
Custom Field
Document Cycle Time
Advanced Packet
Inprocess Packets
Inprocess Advanced Packets
Enhanced Task Dependencies



Task Number	Task Name	Task Status	Workflow Name	Step Name	InfoCard Number	Revision	Verification Required	Validation Required	Response Due Date	Proposed Changes	User ID	Notification Date	Change Description	Infocard Reason For Change
DCO-0026														
	Silicone Systems Approval	In Process	DCO	Collaboration	MD-eForm-0008	02	No	No			MATT	03 Jan 2018 9:47:01 AM		
	Silicone Systems Approval	In Process	DCO	Collaboration	MD-eForm-0009	02	No	No			MATT	03 Jan 2018 9:47:01 AM		
	Silicone Systems Approval	In Process	DCO	Collaboration	Supplier-0014	01	No	No			MATT	03 Jan 2018 9:47:01 AM		
DCO-0029														
	Revise IFU-0001	In Process	DCO	Collaboration	GMP-LBL-0008	02	No	No			PSANDERSON	17 Jan 2018 12:51:19 PM	Section 2.2	Update
	Revise IFU-0001	In Process	DCO	Collaboration	GMP-LBL-0008	02	No	No			DVOLK	17 Jan 2018 11:56:27 AM	Section 2.2	Update
	Revise IFU-0001	In Process	DCO	Collaboration	GMP-LBL-0008	02	No	No			SCHRISTENSEN	17 Jan 2018 12:51:19 PM	Section 2.2	Update
	Revise IFU-0001	In Process	DCO	Collaboration	GMP-LBL-0008	02	No	No			RCARPENTER	17 Jan 2018 12:51:19 PM	Section 2.2	Update
DCO-0030														
	Revise PI-0001	In Process	DCO	Collaboration	GMP-LBL-0007	02	No	No			PSANDERSON	17 Jan 2018 12:49:29 PM	Section 2.2	Warning Updates
	Revise PI-0001	In Process	DCO	Collaboration	GMP-LBL-0007	02	No	No			DVOLK	17 Jan 2018 11:59:54 AM	Section 2.2	Warning Updates
	Revise PI-0001	In Process	DCO	Collaboration	GMP-LBL-0007	02	No	No			SCHRISTENSEN	17 Jan 2018 12:49:29 PM	Section 2.2	Warning Updates
	Revise PI-0001	In Process	DCO	Collaboration	GMP-LBL-0007	02	No	No			RCARPENTER	17 Jan 2018 12:49:29 PM	Section 2.2	Warning Updates
DCO-0031														
	Label Change	In Process	DCO	Collaboration	MD-SOP-0001	04	No	No			DVOLK	24 Jan 2018 11:16:39 AM		
	Label Change	In Process	DCO	Collaboration	MD-SOP-0001	04	No	No			SCHRISTENSEN	24 Jan 2018 11:16:39 AM		
	Label Change	In Process	DCO	Collaboration	MD-SOP-0001	04	No	No			SLISTON	07 Dec 2017 1:27:14 PM		
	Label Change	In Process	DCO	Collaboration	MD-SOP-0001	04	No	No			SKOFAX	25 Jan 2018 4:09:21 PM		
	Label	In Process	DCO	Collaboration	MD-SOP-0001	04	No	No			MATT	24 Jan 2018		


JavaScript Tutorial
https://www.w3schools.com/js/default.asp
HTML CSS JAVASCRIPT SQL PHP BOOTSTRAP HOW TO JQUERY W3.CSS PYTHON MORE REFERENCES EXAMPLES
JS Tutorial
JS HOME
JS Introduction
JS Where To
JS Output
JS Statements
JS Syntax
JS Comments
JS Variables
JS Operators
JS Arithmetic
JS Assignment
JS Data Types
JS Functions
JS Objects
JS Events
JS Strings
JS String Methods
JS Numbers
JS Number Methods
JS Arrays
JS Array Methods
JS Array Sort
JS Array Iteration
JS Dates
JavaScript Tutorial
‹ Home
Next ›
JavaScript is the programming language of HTML and the Web.
JavaScript is easy to learn.
This tutorial will teach you JavaScript from basic to advanced.
Examples in Each Chapter
With our "Try it Yourself" editor, you can change all examples and view the results.
Example
My First JavaScript
Click me to display Date and Time
Try it Yourself »
STARBUCKS® BLONDE FLAT WHITE
ORDER NOW
COLOR PICKER

Real Life Objects, Properties, and Methods

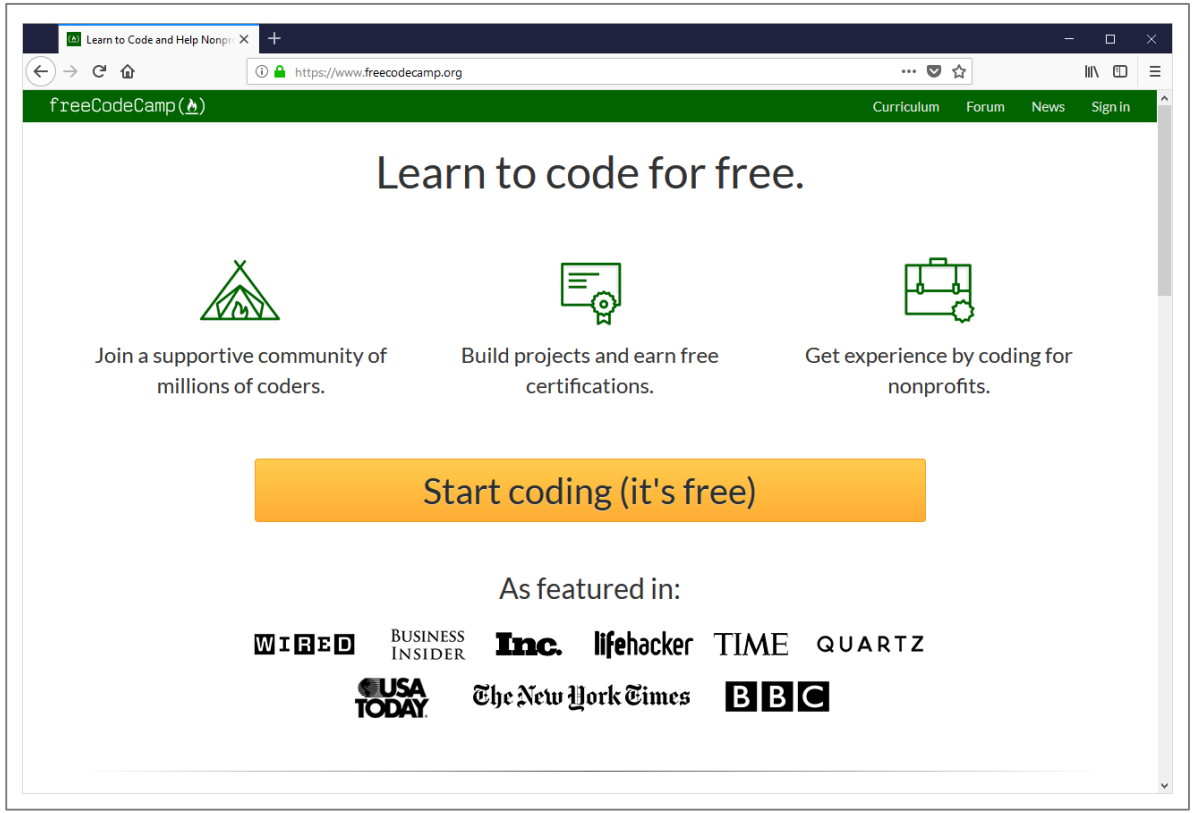
In real life, a car is an **object**.

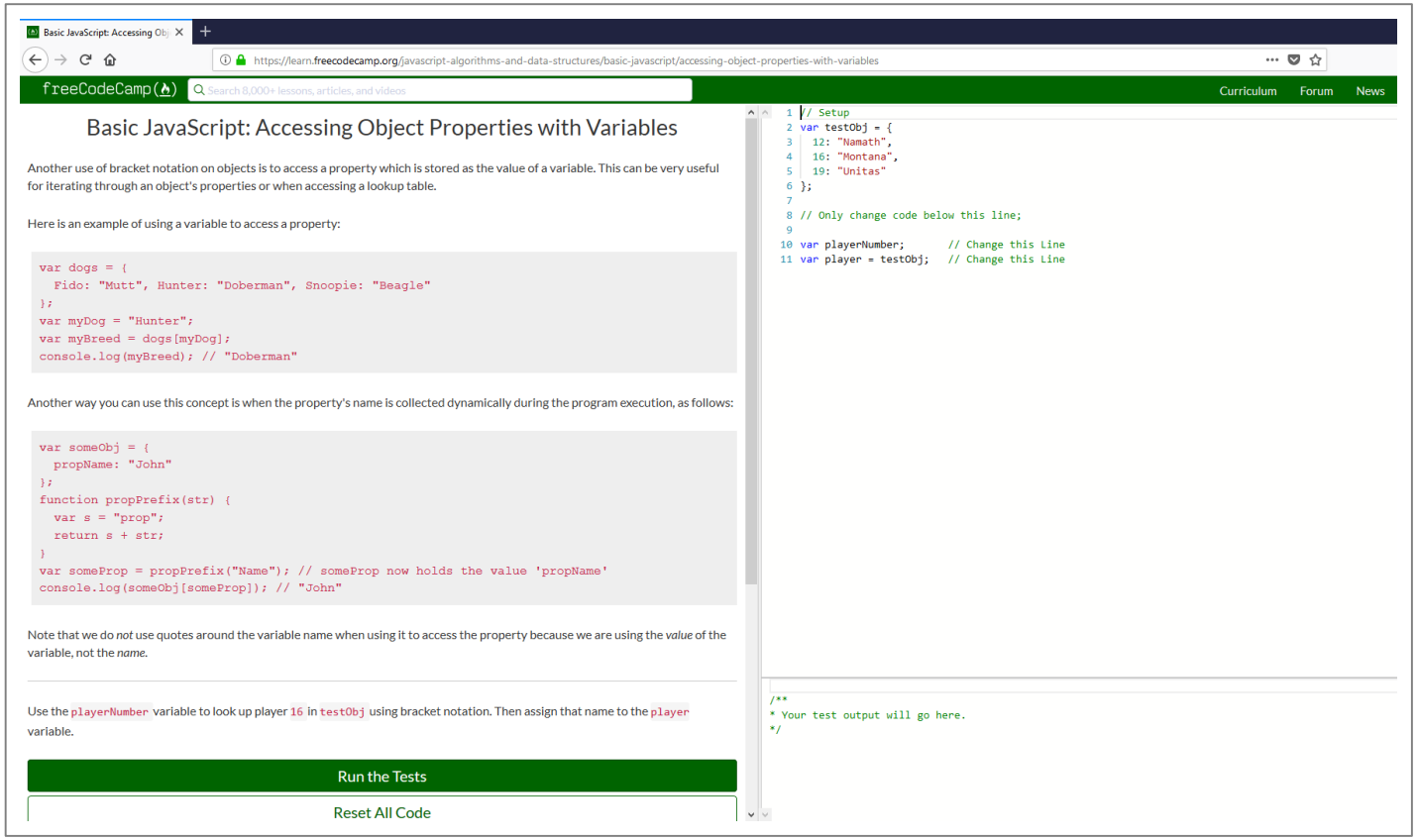
A car has **properties** like weight and color, and **methods** like start and stop:

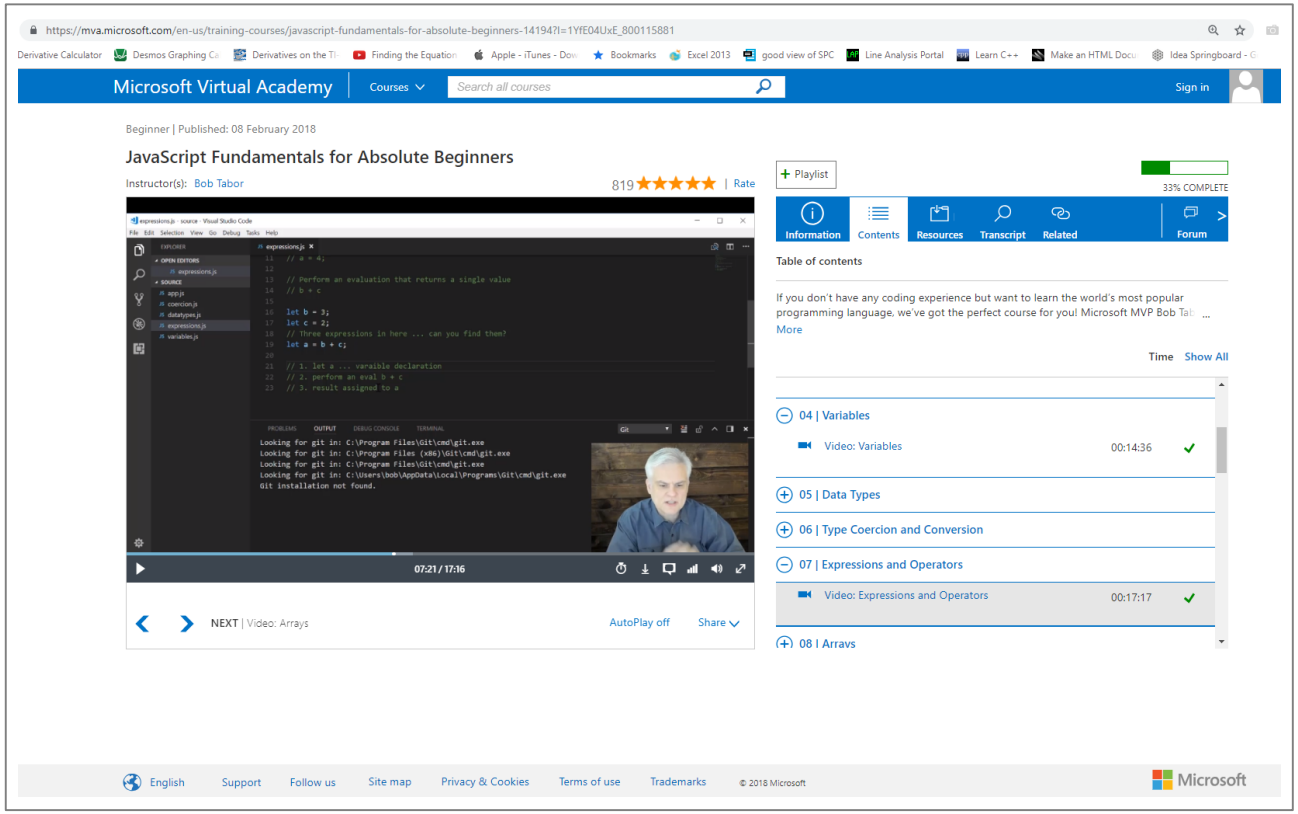
Object	Properties	Methods
	car.name = Fiat car.model = 500 car.weight = 850kg car.color = white	car.start() car.drive() car.brake() car.stop()

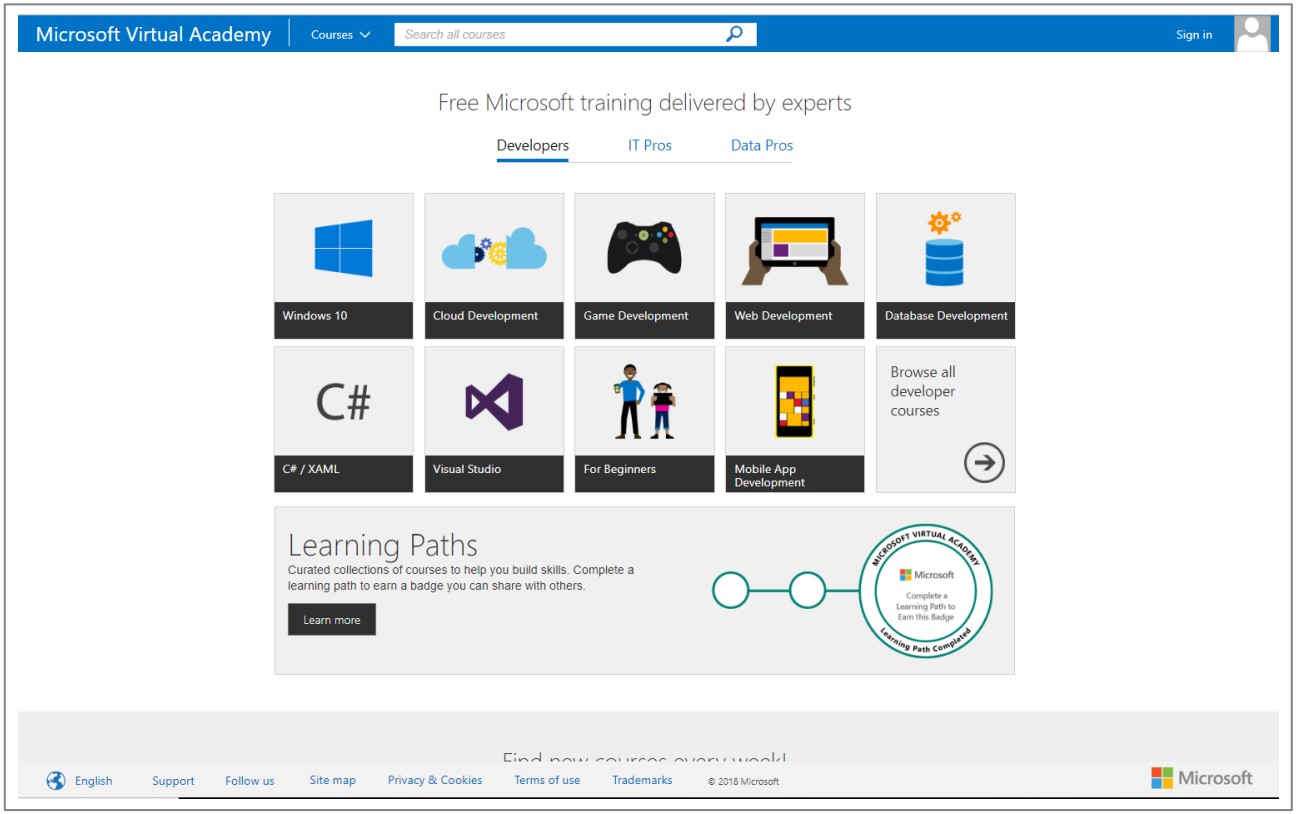
All cars have the same **properties**, but the property **values** differ from car to car.

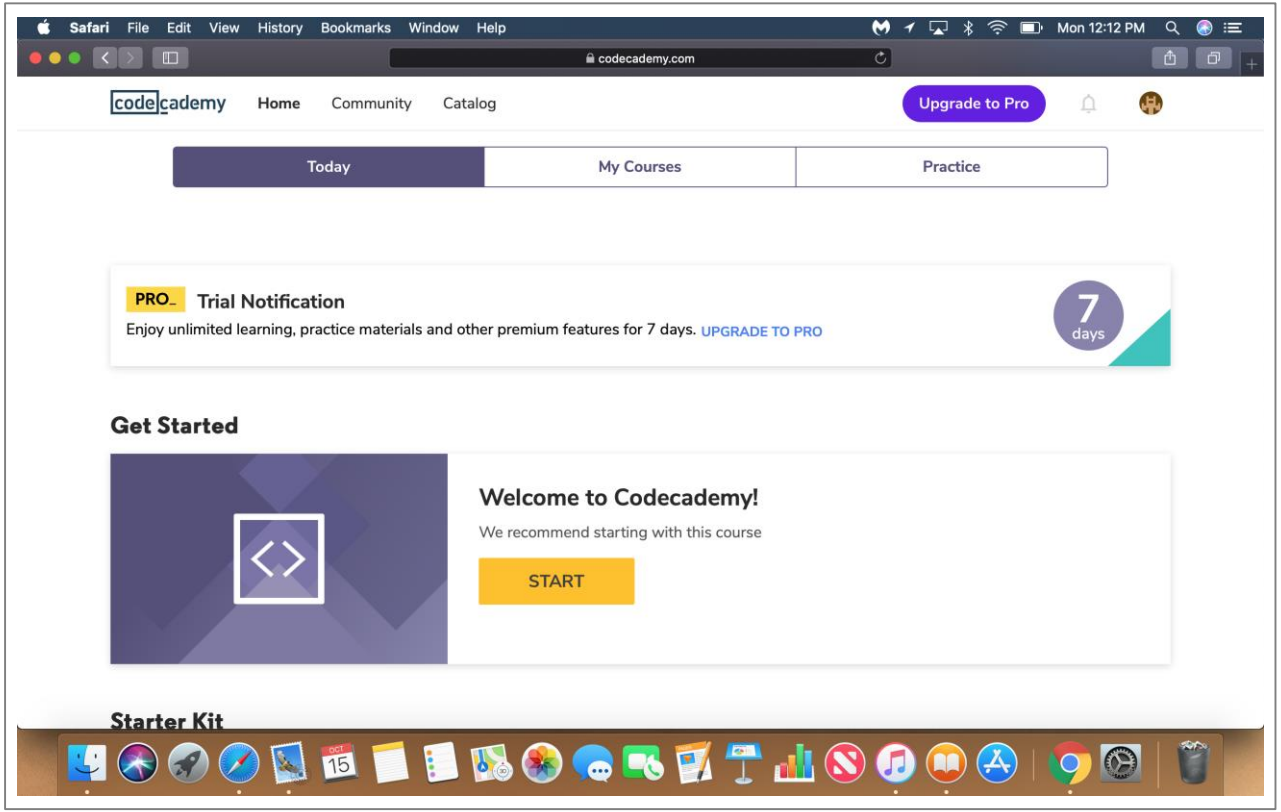
All cars have the same **methods**, but the methods are performed **at different times**.


Learn to Code and Help Nonpr
https://www.freecodecamp.org
freeCodeCamp
Curriculum
Forum
News
Sign in
Learn to code for free.
Join a supportive community of millions of coders.
Build projects and earn free certifications.
Get experience by coding for nonprofits.
Start coding (it's free)
As featured in:
WIRED
BUSINESS INSIDER
Inc.
lifehacker
TIME
QUARTZ
USA TODAY
The New York Times
BBC


Basic JavaScript: Accessing Object Properties with Variables
https://learn.freecodecamp.org/javascript-algorithms-and-data-structures/basic-javascript/accessing-object-properties-with-variables
freeCodeCamp
Search 8,000+ lessons, articles, and videos
Curriculum
Forum
News
Basic JavaScript: Accessing Object Properties with Variables
Another use of bracket notation on objects is to access a property which is stored as the value of a variable. This can be very useful for iterating through an object's properties or when accessing a lookup table.
Here is an example of using a variable to access a property:
var dogs = {
Fido: "Mutt", Hunter: "Doberman", Snoopie: "Beagle"
};
var myDog = "Hunter";
var myBreed = dogs[myDog];
console.log(myBreed); // "Doberman"
Another way you can use this concept is when the property's name is collected dynamically during the program execution, as follows:
var someObj = {
propName: "John"
};
function propPrefix(str) {
var s = "prop";
return s + str;
}
var someProp = propPrefix("Name"); // someProp now holds the value 'propName'
console.log(someObj[someProp]); // "John"
Note that we do not use quotes around the variable name when using it to access the property because we are using the value of the variable, not the name.
Use the playerNumber variable to look up player 16 in testObj using bracket notation. Then assign that name to the player variable.
Run the Tests
Reset All Code
// Setup
var testObj = {
12: "Namath",
16: "Montana",
19: "Unitas"
};
// Only change code below this line;
var playerNumber; // Change this Line
var player = testObj; // Change this Line
/**
* Your test output will go here.
*/


https://mva.microsoft.com/en-us/training-courses/javascript-fundamentals-for-absolute-beginners-14194?l=1YfE04UxE_800115881
Microsoft Virtual Academy
Courses
Search all courses
Sign in
Beginner | Published: 08 February 2018
JavaScript Fundamentals for Absolute Beginners
Instructor(s): Bob Tabor
819
Rate
+ Playlist
33% COMPLETE
Information
Contents
Resources
Transcript
Related
Forum
Table of contents
If you don't have any coding experience but want to learn the world's most popular programming language, we've got the perfect course for you! Microsoft MVP Bob Tab ...
More
Time
Show All
04 | Variables
Video: Variables
00:14:36
05 | Data Types
06 | Type Coercion and Conversion
07 | Expressions and Operators
Video: Expressions and Operators
00:17:17
08 | Arrays
07:21 / 17:16
NEXT | Video: Arrays
AutoPlay off
Share
English
Support
Follow us
Site map
Privacy & Cookies
Terms of use
Trademarks
© 2018 Microsoft
Microsoft


Microsoft Virtual Academy
Courses
Search all courses
Sign in
Free Microsoft training delivered by experts
Developers
IT Pros
Data Pros
Windows 10
Cloud Development
Game Development
Web Development
Database Development
C#
C# / XAML
Visual Studio
For Beginners
Mobile App Development
Browse all developer courses
Learning Paths
Curated collections of courses to help you build skills. Complete a learning path to earn a badge you can share with others.
Learn more
MICROSOFT VIRTUAL ACADEMY
Microsoft
Complete a Learning Path to Earn this Badge
Learning Path Completed
English
Support
Follow us
Site map
Privacy & Cookies
Terms of use
Trademarks
© 2018 Microsoft
Microsoft


Safari File Edit View History Bookmarks Window Help
Mon 12:12 PM
codecademy.com
codecademy
Home
Community
Catalog
Upgrade to Pro
Today
My Courses
Practice
PRO_ Trial Notification
Enjoy unlimited learning, practice materials and other premium features for 7 days. UPGRADE TO PRO
7 days
Get Started
Welcome to Codecademy!
We recommend starting with this course
START
Starter Kit

References

- (n.d.). Retrieved from https://www.w3schools.com/js/default.asp
- Learn to code with free online courses, programming projects, and interview preparation for developer jobs. (n.d.). Retrieved from https://www.freecodecamp.org/
- Microsoft Virtual Academy. (n.d.). Retrieved from https://mva.microsoft.com/
- Learn to code - for free. (n.d.). Retrieved from https://www.codecademy.com/
- User Guide & Tutorial – JReport. (n.d.). Retrieved from https://www.jinfonet.com/documentation/