# Introduction to JavaScript – Part One

Website Development 2

# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# Introduction to JavaScript

- JavaScript is used in millions of Web pages to improve the design, validate forms, detect browsers, create cookies, and much more.
- JavaScript is the most popular scripting language on the internet, and works in all major browsers, such as Chrome, Internet Explorer, Mozilla, Firefox, Netscape, Opera.

# Introduction to JavaScript

- JavaScript was designed to add interactivity to HTML pages.
- JavaScript is a scripting language (a scripting language is a lightweight programming language).
- A JavaScript consists of lines of executable computer code.
- A JavaScript is usually embedded directly into HTML pages.
- JavaScript is an interpreted language (means that scripts execute without preliminary compilation).
- Everyone can use JavaScript without purchasing a license.

# Are Java and JavaScript the Same?

- NO!
- Java and JavaScript are two completely different languages in both concept and design!
- Java (developed by Sun Microsystems) is a powerful and much more complex programming language - in the same category as C and C++.

# How to Put a JavaScript into an HTML Page?

- Use the script tag to enclose the code.

```
<script>
   document.write("Hello World!")
</script>
```

# Ending Statements With a Semicolon?

- With traditional programming languages, like C++ and Java, each code statement has to end with a semicolon (;).
- Many programmers continue this habit when writing JavaScript, but in general, semicolons are **optional**! However, semicolons are required if you want to put more than one statement on a single line.

# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# Variables

- Variables are used to store data.
- A variable is a "container" for information you want to store.  A variable's value can change during the script. You can refer to a variable by name to see its value or to change its value.

```
let myname;
let num1=3;
let num2=4;
```

# Variables

- Rules for variable names:
  - Variable names are case sensitive
    - strname – STRNAME (not the same).
  - They must begin with a letter, $,  or the underscore character.
  - Can contain numbers, letters, $,  and the underscore character.
  - No spaces allowed.

# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# JavaScript Operators – 1

## Arithmetic Operators

Arithmetic operators are used to perform arithmetic on numbers (literals or variables).

total=num1 * num2

Taking our previous values for num1 and num2, after execution of the above statement → total = 12.

| Operator | Description | Example | Result |
|---|---|---|---|
| + | Addition | x=2<br>y=2<br>x+y | 4 |
| - | Subtraction | x=5<br>y=2<br>x-y | 3 |
| * | Multiplication | x=5<br>y=4<br>x*y | 20 |
| / | Division | 15/5<br>5/2 | 3<br>2,5 |
| % | Modulus (division remainder) | 5%2<br>10%8<br>10%2 | 1<br>2<br>0 |
| ++ | Increment | x=5<br>x++ | x=6 |
| -- | Decrement | x=5<br>x-- | x=4 |

# JavaScript Operators – 2

## Assignment Operators

Assignment operators assign values to JavaScript variables.

| Operator | Example | Is The 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 |

# JavaScript Operators – 3

## Comparison Operators

This table contains the different comparison operators.

When checking if two values are equal, it is considered better to use strict equals operators (===) and (!==) rather than (==) and (!=) as these strict operators

check that the value and data types match.

| Operator | Description | Example |
|---|---|---|
| == | is equal to | 5==8 returns false |
| === | is equal to (checks for both value and type) | x=5<br>y="5"<br><br>x==y returns true<br><br>x===y returns false |
| != | is not equal | 5!=8 returns true |
| !== | Is not identical | 4!==5 (true)<br>5!==5 (false)<br>5!=='5' (true) |
| > | is greater than | 5>8 returns false |
| < | is less than | 5<8 returns true |
| >= | is greater than or equal to | 5>=8 returns false |
| <= | is less than or equal to | 5<=8 returns true |

# JavaScript Operators – 4

## Logical Operators

To further enhance your **if** statements you can use the so-called **logical** operators.

| Operator | Description | Example |
|---|---|---|
| && | and | x=6<br>y=3<br>(x < 10 && y > 1) returns true |
| \|\| | or | x=6<br>y=3<br>(x==5 \|\| y==5) returns false |
| ! | not | x=6<br>y=3<br>!(x==y) returns true |

# JavaScript Basic Examples

```
<script>
 document.write("Hello World!");
</script>
```

```
<script>
 document.write("<b>Hello World!</b>");
</script>
```

# JavaScript Basic Examples

```
<script>
 let x="Hello World!";
 document.write(x)
</script>
```

```
<script>
 let x="Joe Bloggs";
 document.write("Good morning" +x);
</script>
```

# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# JavaScript Popup Boxes – 1

- **Alert Box**
  - An alert box is often used if you want to make sure information comes through to the user.
  - When an alert box pops up, the user will have to click "OK" to proceed.

```
<script>
 alert("Press OK to continue!")
</script>
```

# JavaScript Popup Boxes – 2

- **Confirm Box**
  - A confirm box is often used if you want the user to verify or accept something.
  - When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed.
  - If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

# JavaScript Popup Boxes – 3

- **Prompt Box**
  - A prompt box is often used if you want the user to input a value.
  - When a prompt box pops up, the user will have to click either "OK" or "Cancel" to proceed after entering an input value.
  - If the user clicks "OK“, the box returns the input value. If the user clicks "Cancel“, the box returns null.

# Prompt Box Example

```
<script>
 let yname=prompt ("Enter your name", " ");
 document.write("Welcome ",+ yname)
</script>
```

# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# Global methods

- **isNaN(value)**
  - This function returns true if its argument is not a number and false if it is numeric.

```
<script>
  let yournum=prompt("Enter a number between 1 and 10", "");
  if (isNaN(yournum)) {
    alert("This is not a number");
</script>
```

# Global methods

- **parseInt(string [, radix])**
  - The string is parsed and its value as an integer returned. Once an invalid character is encountered the parsing stops and the function returns what it has already found. If the first character of the string is invalid, **NaN** is returned.

```
let yournum=parseInt(prompt("Enter a number between 1 and 10", ""));
```

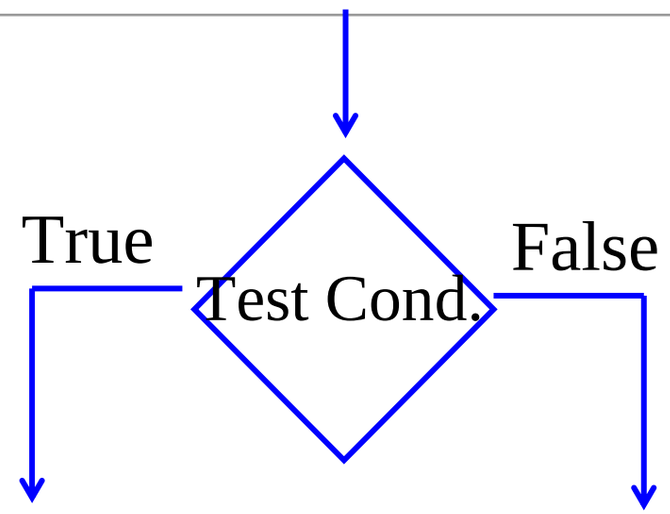
- **parseFloat(string)**
  - This function parses a string, passed in as an argument, and returns it as a floating point number. Once an invalid character is encountered the parsing stops and the function returns what it has already found. If the first character of the string does not belong to the valid set, **NaN** is returned.

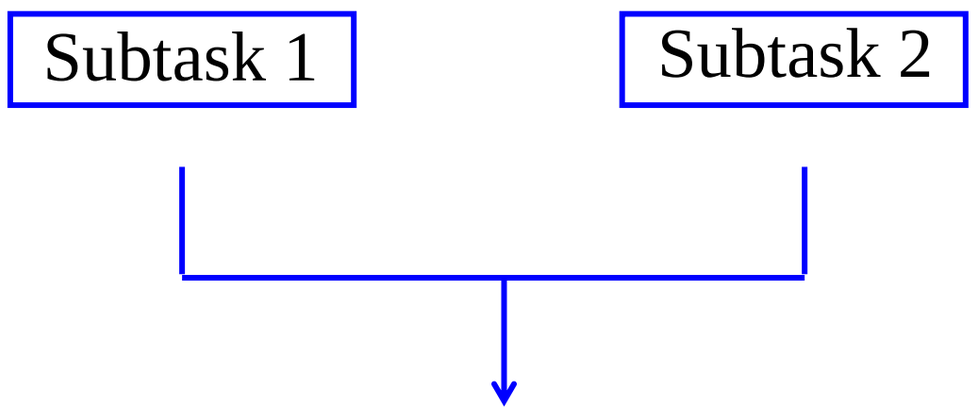
# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# 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.



# Conditional Statements

- In JavaScript we have the following conditional statements:
  - **if statement** - use this statement if you want to execute some code only if a specified condition is true.
  - **if...else statement** - use this statement if you want to execute some code if the condition is true and another code if the condition is false.
- **if...else if....else statement** - use this statement if you want to select one of many blocks of code to be executed.
- **switch statement** - use this statement if you want to select one of many blocks of code to be executed.

# Conditional Statements - syntax

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

---

```
if (condition)
{
code to be executed if condition is true
}
else
{
code to be executed if condition is not true
}
```

# Conditional Statements Examples – 1

```
<script>
 let x=3
 if(x<0)
 {
  alert ("negative value");
 }
 else
 {
  alert ("positive value");
 }
</script>
```

# Conditional Statements Examples – 2

```
<script>
 let ans=confirm("You are ready to proceed with the order?");
 if(ans)
 {
  alert ("Get your credit card");
 }
 else
 {
  alert ("The shopping cart will be emptied");
 }
</script>
```

# Conditional Statements Examples – 3

```
<script>
 let y=prompt("Which year are you in?", " ");
 if(y===1)
 {
   alert("Welcome to WIT");
 }
 else
 {
  alert("Welcome back to WIT");
 }
</script>
```

# Conditional Statements Examples – 4

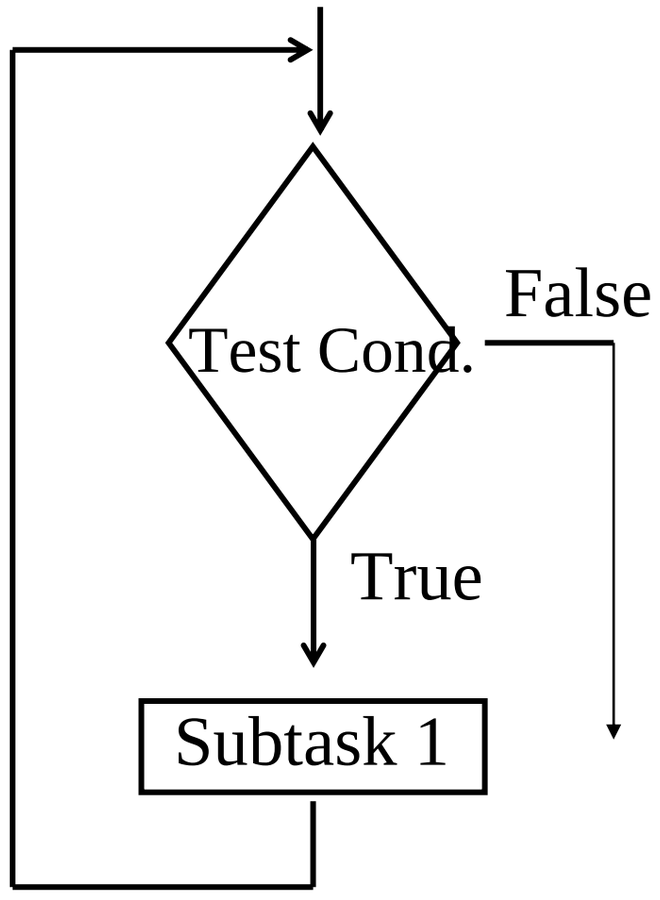
```
<script>
 let grade='A';
 switch (grade) {
 case 'A': document.write("Good job"); break;
 case 'B': document.write("Pretty good"); break;
 case 'C': document.write("Passed"); break;
 case 'D': document.write("Not so good"); break;
 case 'F': document.write("Failed"); break;
 default: document.write("Unknown grade") }
</script>
```

# Lecture Outline

- Introduction to JavaScript
- Variables
- Operators
- Pop Up Boxes
- Global Methods
- Conditional Statements
- Iterative Statements

# Iterative statements

- Very often when you write code, you want to perform the same actions a number of times. You can use iterative statements in your code to do this.



# Iterative statements

- There are two different kinds of loops: **for** and **while**.
- The **for** loop is used when you know in advance how many times the script should perform.
- The **while** loop is used when you want the loop to continue until a certain condition becomes true.

# Iterative statements examples – 1

- **For Loop** example, which displays the numbers from 5 down to (and including) 1

```
<script>
 for (i = 5; i > 0; i--)
  {
    document.write( i + "<br>");
  }
</script>
```

# Iterative statements examples – 2

- **While Loop** example, which also displays the numbers from 5 down to (and including) 1

```
<script>
 let i=5;
 while(i > 0)
 {
      document.write( i + "<br>");
      i--;
 }
</script>
```

# Iterative statements examples – 3

- **Do … While** Loop example, which executes until the number 9 is entered by the user

```
<script>
 let ans=9;
 do {
   yournum=parseInt(prompt("Enter a number between 1 and 10", ""));
  }while(ans!=yournum);
</script>
```