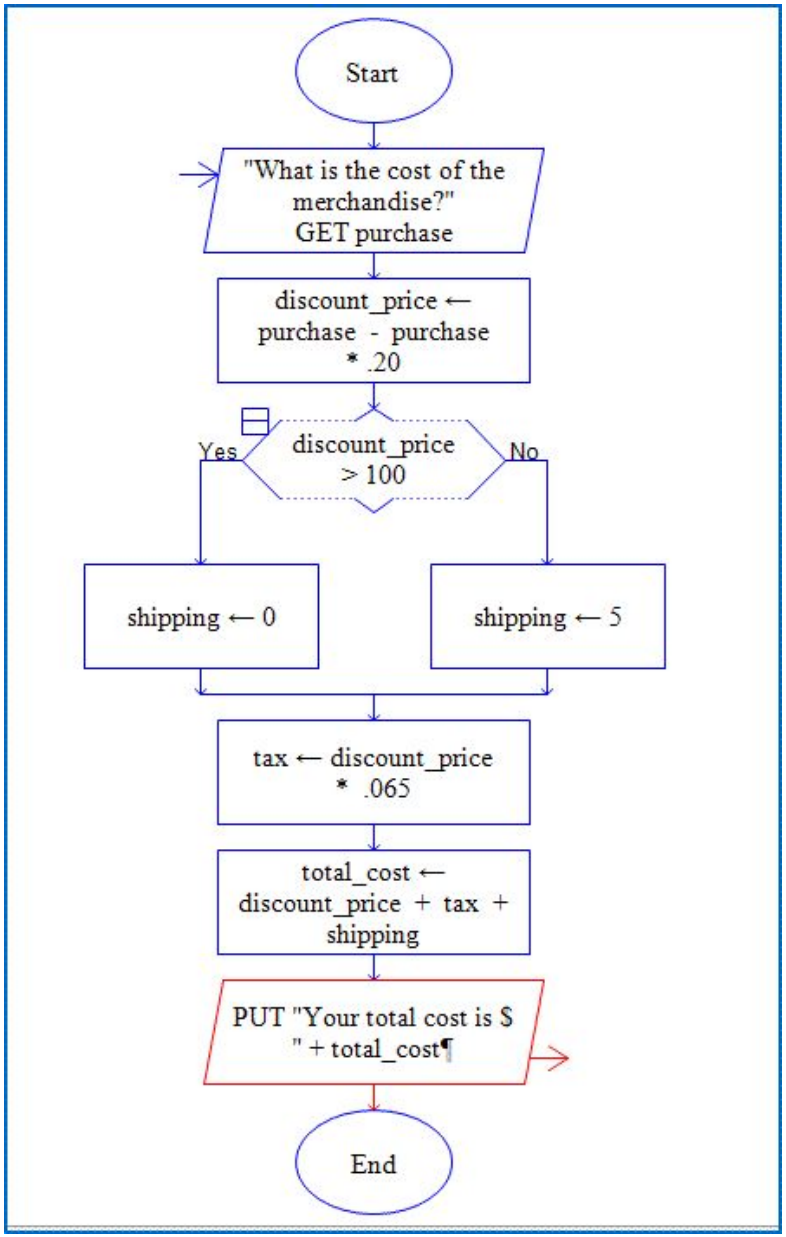
JavaScript Programming Basics

1.1

What is Programming?

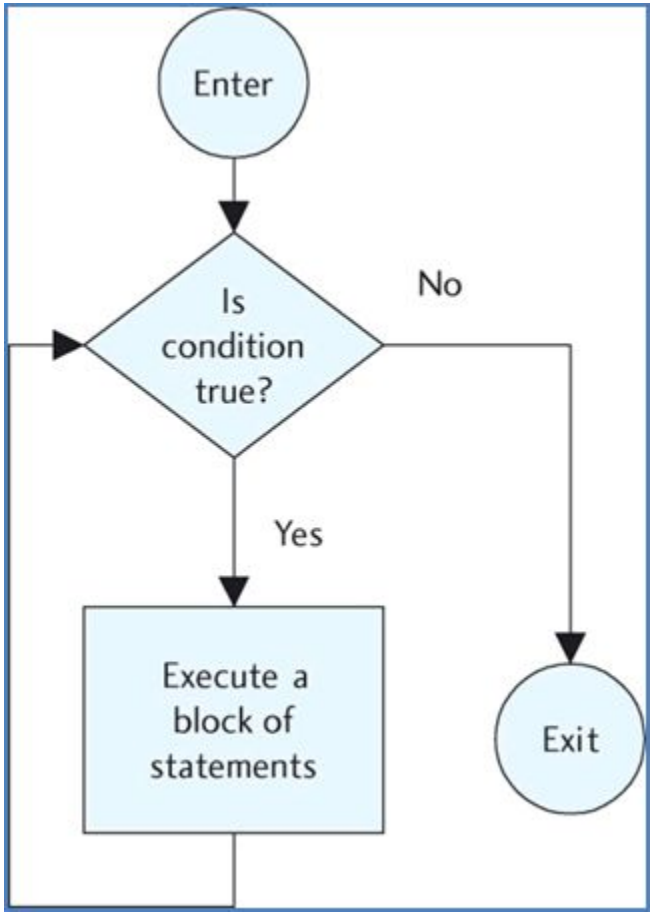


The Program Development Cycle

- Analyze the problem
  - What information are you given?
  - What information is needed to get desired results?
- Design a program to solve the problem
  - Include step-by-step instructions
  - Can use flowcharts or pseudocode
- Code the program
- Test the program

At any time you find a flaw, revise!

1.2 The Structure of a Program



Input-Processing-Output

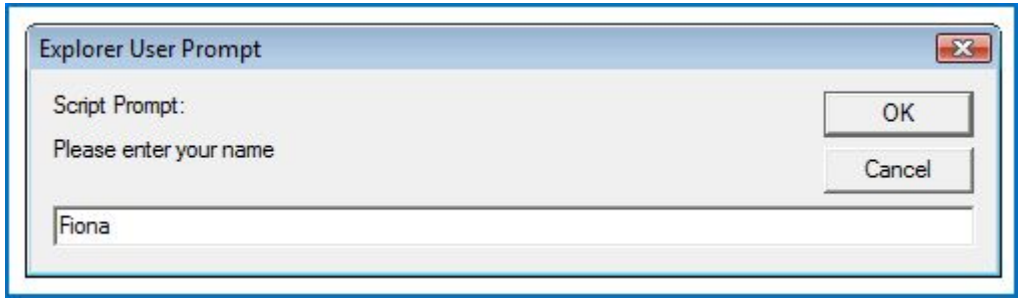
- Input:
  - prompts
  - previous values
  - other files
- Processing:
  - what the program does with input and other information
- Output:
  - display on screen
  - information sent to other places

Prompts

Prompt:

```
var name = prompt("Please enter your name"," ");
```

User sees:



(Internet Explorer)

(Firefox)

After entering "`Fiona`", the variable `name` holds the value "`Fiona`"

Processing the Input

Example:

```
<script type="text/javascript">
    var name = prompt("Please enter your name"," ");
    var greeting = "Hello there, " + name + "!";
</script>
```

If the user enters "Fiona", the variable `name = "Fiona"` and the variable `greeting = "Hello there, Fiona!"`

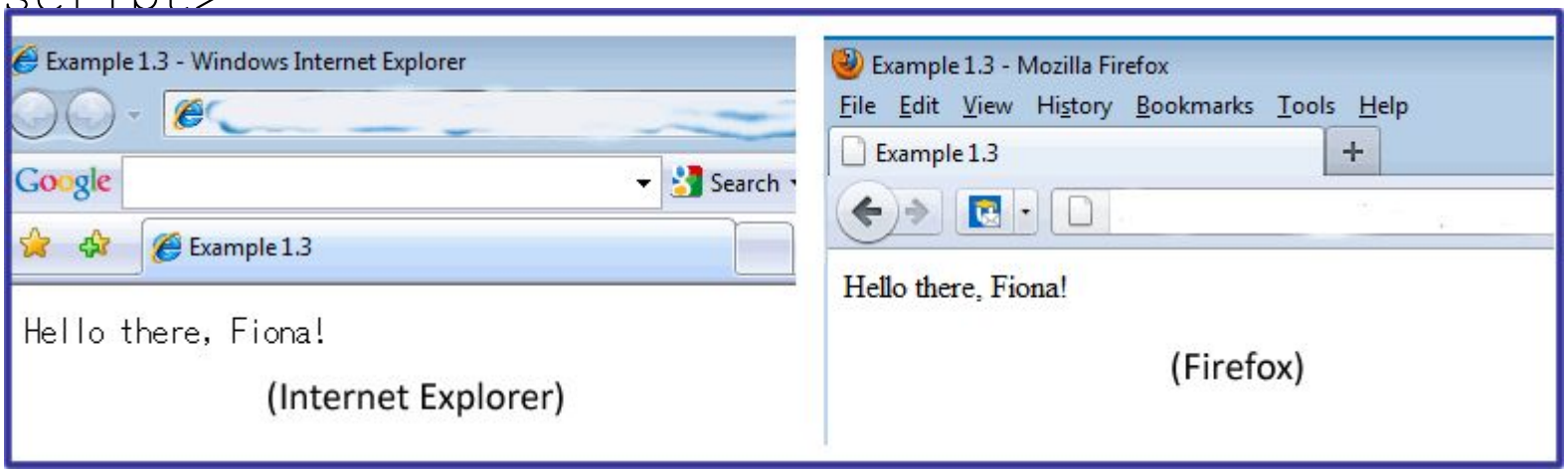
Output

Example:

```
<script type="text/javascript">
    var name = prompt("Please enter your name"," ");
    var greeting = "Hello there, " + name + "!";

    document.write(greeting);

</script>
```



The Control Structures

The sequential (or sequence) structure

statements execute in sequence

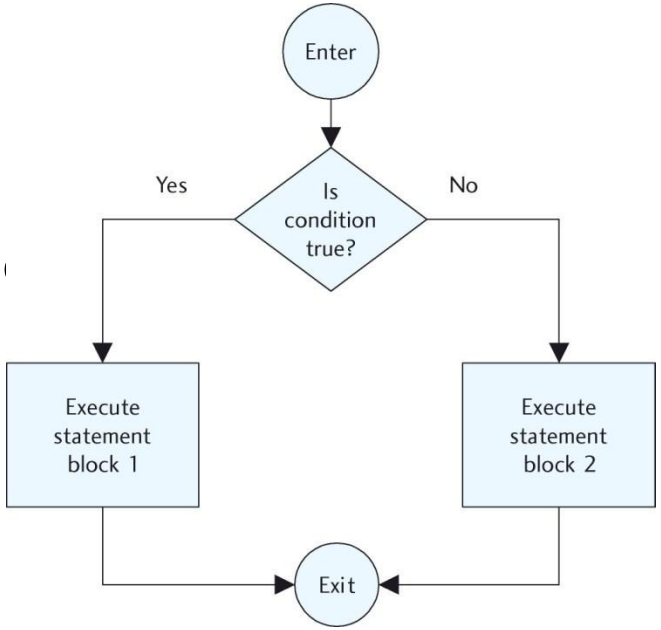
The decision (or selection) structure

statements execute if a condition is tr u

if not, either nothing happens or other

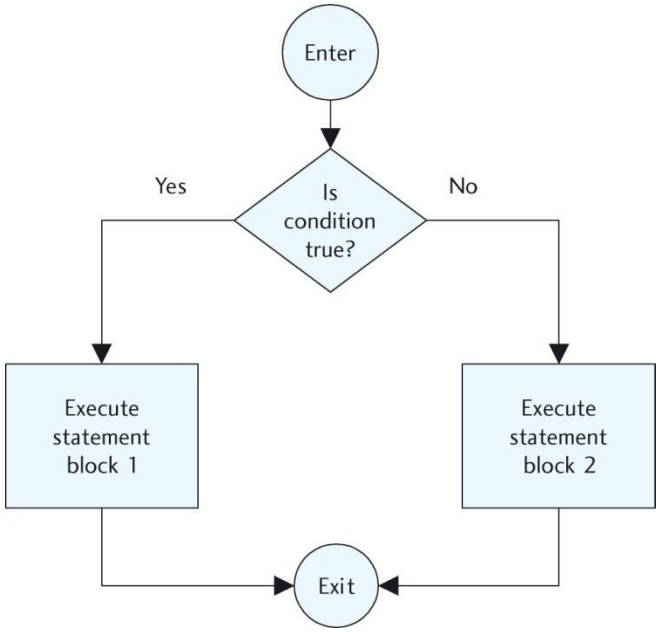
The loop (or repetition) structure

statements execute until a condition is

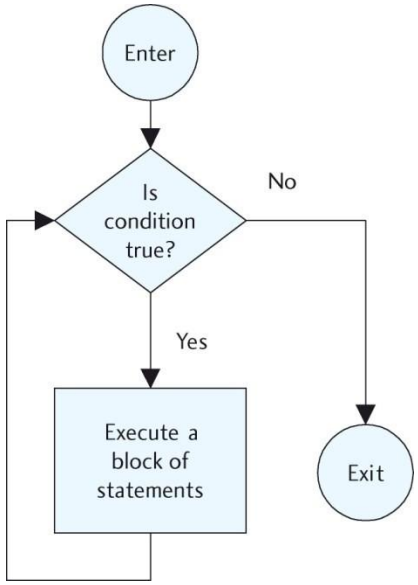


The Control Structures

The decision (or selection) structure



The loop (or repetition) structure



1.3
Data Types and Operations on Data

Numerical Data

- Numbers are values that can be processed and calculated.
- Many languages make a distinction between integers and floating point numbers.
- JavaScript: when a number is stored in a variable, it is initially treated as a floating point number.
- All numbers in JavaScript are initially stored as the **numerical** data type.
- When a number is entered into a prompt box, it is initially stored as a **text** value.
  - It cannot be used in a calculation.
  - It must be turned into a numeric value to use in a calculation.

String Data

- Strings are a series of keyboard characters enclosed in quotation marks.
- Strings can consist of words, phrases, sentences, and even whole paragraphs.
- A string can also be a single character such as a letter or a punctuation character.
- When a number is stored as a string, it cannot be used in a numerical calculation or process.

Variables and Named Constants

- A variable is called a variable because it can vary.
- A quantity that can change value during the execution of a program.
- Any time we need to refer to that data, we refer to its **variable name**.
- A named constant is a value that is used often in a program but will not change value throughout the program, such as the number in a dozen or the tax rate charged on purchases.

Assignment Statements

Declaring variables: Use the `var` keyword

```
var age;
```

creates a variable named `age`

```
var age = 23;
```

creates a variable named **`age`** which is assigned an initial value of `23`

Operations on Data

Arithmetic Operations

Operator	Description	Example	Result, if y = 3
+	Addition	x = y + 2	x = 5
-	Subtraction	x = y - 2	x = 1
*	Multiplication	x = y * 2	x = 6
/	Division	x = y / 2	x = 1.5
%	Modulus	x = y % 2	x = 1

The Concatenation Operator

- Concatenation Operator: joins two strings together
- The symbol is **+** but, by the context, the computer knows that it is not used to add values.

Example:

```
greeting = "Good morning"

name = "Robbie"
```

The following statement concatenates the variables and other text and stores it as one string in a third variable named **welcome**:

```
welcome = greeting + ", " + name;
```

After the execution of this statement, the variable **welcome** contains :

```
"Good morning, Robbie"
```

1.4

Problem Solving: The Importance of Logical Thinking

Pseudocode and Flowcharts

Pseudocode is a way to think through and design a program before writing the actual code.

- Pseudocode uses short, English-like phrases to describe the outline of a program.
- Flowcharts are diagrams that use special symbols to display pictorially the flow of execution within a program or program module.
- Often programmers use both pseudocode and flowcharts at various stages of a program's development.

1.5 JavaScript in the Web Page

The <script></script> Tag Pair

The <noscript></noscript> Tag Pair

<script></script>:

Used to define a client-side script like JavaScript

<noscript></noscript>:

Used to provide alternate content for users who have disabled scripts in their browsers

Used to provide alternate content for browsers that don't support client-side scripting – rare today

```
<noscript>
    Sorry, your browser doesn't support JavaScript.
</noscript>
```

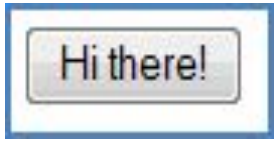
JavaScript in a Web Page <body>

Using inline JavaScript with a button:

Code to add a button to a web page:

```
<input type="button" id="myButton" value="Hi there!"
      onclick="alert('Well, hello my friend.');" />
```

Creates a button that looks like this:



When clicked, you get an alert that says:

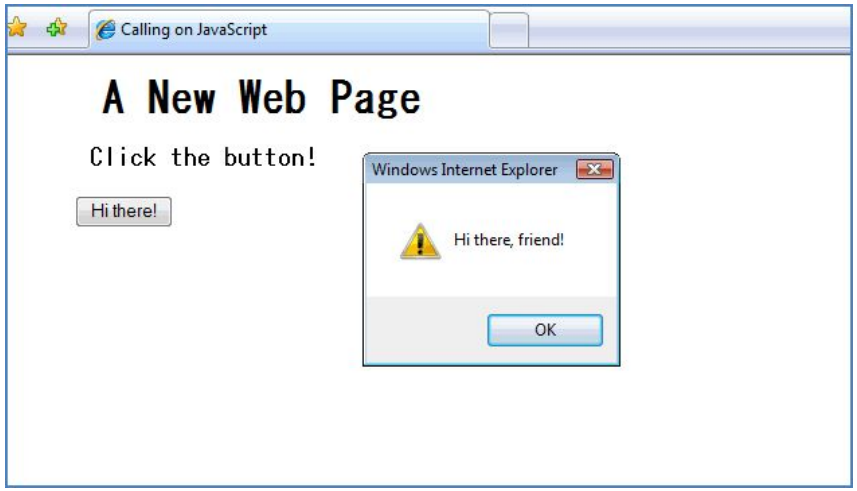
```
Well, hello, my friend.
```

JavaScript in the Document <head> Section

Most JavaScript we will write will be in <head> section

JavaScript executes when user does something in web page

```
<html>
<head>
<title>Example</title>
<script>
function welcome()
{
    alert("Hi there, friend!");
}
</script>
</<head>>
<body>
   <h1>A New Web Page</h1>
   <h3>Click the button! </h3>
   <p><input type="button"
        id="myButton" value="Hi
        there!" onclick =
        "welcome();" /></p>
</body>
</html>
```



The <body> onload Event

Loads JavaScript before user views the page, as it is loading

```
<html>
<head>
<title>Example</title>
<script>
function welcome()
{
    alert("Hi there, friend!");
}
</script>
</<head>>
<body onload = "welcome()">
   <h1>A New Web Page</h1>
   <h3> This site is nothing but fun!</h3>
</body>
</html>
```

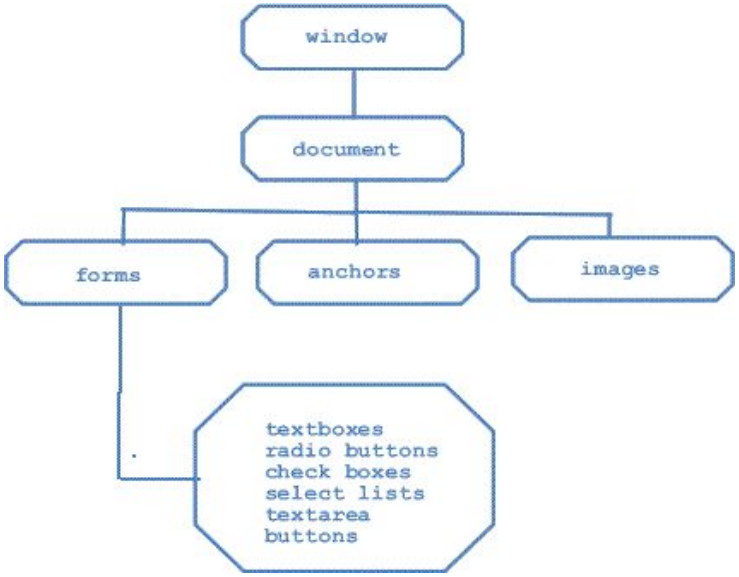
1.6 Introduction to Objects

What is an Object? Properties and Methods

- Anything that has properties and a function (or functions) is an **object**.
- Properties are qualities, traits, or attributes common to a specific thing—or object.
  - Properties (also called attributes) describe the object
- A function, in this context, is a process or operation executed by or to the object.
  - Methods (also called functions) are the things the object can do or have done to it.

The Document Object

- An HTML document is an object.
- It uses the **Document Object Model** (**DOM**)



Dot Notation

- You instruct the browser where to place content by using **dot notation**.
- The object is accessed, then a dot, and then any further instructions (methods or attributes) are appended.

```
document.write("Welcome to my first JavaScript page!");
```

The `document` object is accessed. The dot says to use the `write()` method on the `document` object.

The `getElementById()` Method

- Each part of a web page is called an element
- To access an element use the `getElementById()` method
- Allows access a particular container within a document
- Each container must be marked with an identifier
- Add an `id` attribute to the HTML tag

The `innerHTML` Property

The `innerHTML` property sets or returns the inner HTML of an element.

```
1.  <html>
2.  <head>
3.  <title>Example</title>
4.  <script type="text/javascript">
5.  function getValue()
6.  {
7.        var dog = document.getElementById("puppy");
8.        document.write("Your dog is not a terrier <br
          />");  document.write("It is a ");
9.
10.       document.write(dog.innerHTML);
11. }
12.</script
    >
13.</head>
14.<body>
15.       <h1 id = "puppy" onclick="getValue()">Poodle</h1>
16.</body>
17.</html
    >
```

- The `id` of the `<h1>` tag is "`puppy`" (line 15).
- Line 6 gets the value of the contents of the element with the `id` "`puppy`".
- Line 10 uses that value in the output

The `open()` and `close()` Methods

These methods open or close a new window

The `innerHTML` property sets or returns the inner HTML of an element.

```
<html>
<head>
<title>Using the open() and close() Methods</title>
<script type="text/javascript">
function openWin()
{
        smallWindow = window.open("","", "width=300, height=200");
        smallWindow.document.write("<p>Hi again, old friend!<br />Glad to see
                                    you today</p>");
}
function closeWin()
{
 smallWindow.close();
}
</script>
</head>
<body>
        <input type = "button" value = "Open a small window" onclick
                        = "openWin()" />
        <input type = "button" value = "Close the small window"
                        onclick = "closeWin()" />
</body>
</html>
```

1.7 Introduction to JavaScript Functions and Events

JavaScript Functions

- A function is used to isolate a group of instructional statements so that other parts of the program can use that code.
- Functions and methods can normally be used interchangeably.
- Two main categories of functions: user-created and built-in
- Syntax to create your own function:
  - type the `function` keyword
  - Follow with the function's name
  - Put all statements within opening and closing curly brackets ({ }).

```
function name()
{
        JavaScript statements...;
}
```

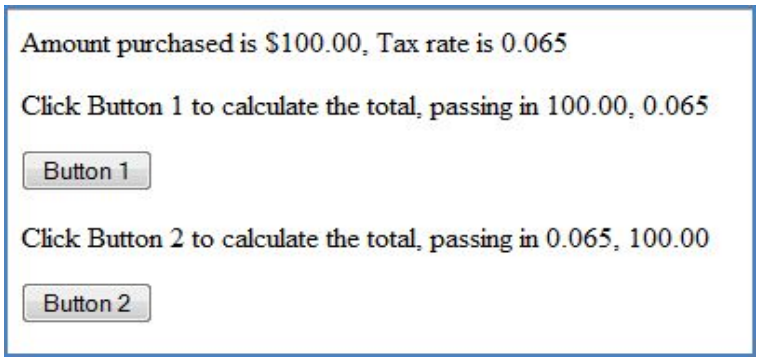
Built-in Functions

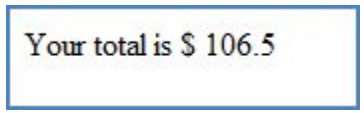
- Some built-in JavaScript functions that we have used so far:
  - `alert()`
  - `write()`
  - `open()`
  - `close()`
  - `getElementById()`

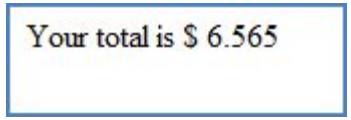
Parameters

In general, parameters are values that are passed into a function.

```
<head>
<title>Using parameters</title>
<script type="text/javascript">
function calculateTotal(purchaseAmt, taxRate)
{
   tax = purchaseAmt * taxRate;
   total = purchaseAmt + tax;
   document.write("Your total is $ "+total);
}
</script>
</head>
<body>
<p>Amount purchased is $100.00, Tax rate is
0.065</p>
<p>Click Button 1 to calculate total, passing
in 100.00, 0.065<p>
<input type="button" value="Button 1"
onclick
= "calculateTotal(100, .04)" />
<p>Click Button 2 to calculate the total,
passing in 0.065, 100.00<p>
<input type="button" value="Button 1"  onclick
= "calculateTotal(0.065, 100)" />
</body>
```







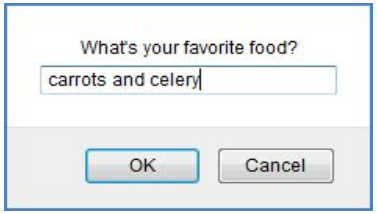
The `prompt()` Function

Allows us to prompt the user to input values which can then be used in any way

```
<head>
<title>The prompt()
Function</title>
<script
type="text/javascript">
function showPrompt()
{
        var food = prompt("What's
your  favorite  food?",  "carrots
and  celery");
        document.write("It's your
lucky day! " + food +" is on
today's  lunch menu!");
}
</script>
</head>
<body>
<input type="button" onclick =
"showPrompt()" value = "push me"
/>
</body>
```

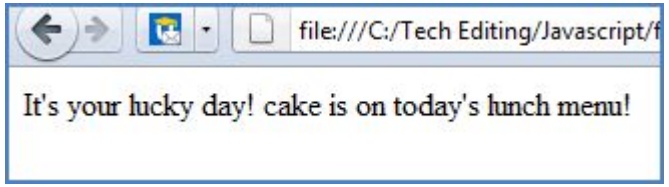
You first see:

After pushing the button:



If user enters pizza:

If user enters cake:



Introduction to JavaScript Events

- An **event** is an action that can be detected by JavaScript
- Usually events are used in combination with functions
- When an event occurs, the function is executed
- Called event-driven programming
- Events:
  - a mouse click
  - a web page or image loading
  - rolling a mouse over a link, an image, or another hot spot on a web page
  - selecting an element or a field on a form

Using a Prompt and an Event

```
 1.    <html>
 2.    <head>
 3.    <title> JavaScript Events</title>
 4.    <script type="text/javascript">
5.function greet()  6.
  {
 7.      var name = prompt("Please enter your name"," ");
8.document.write("<h2>Hello " + name + "! <br />How  are
  you today?</h2>");
  9. }
10.    </script>
11.    </head>
12.    <body>
13.    <h2 id ="hello">Who are you?</h2>
14.    <button type="button" onclick="greet()">Enter your
  name</button>
15.    </body>
16.    </html>
```

Example Output

Initially, this page has a single line and a button and looks like this:

If the user presses the button, types in `Helmut Lindstrom` at the prompt, and presses OK, the page will now look like this: