

# Introduction to JavaScript Training

## Conditionals and Loops

## Lesson 1, Activity 2: Conditionals

There are two types of conditionals in JavaScript.

1. `if - else if - else`
2. `switch / case`

### if - else if - else Conditions

The structure of an `if - else if - else` condition is shown below:

```
if (conditions) {
  statements;
} else if (conditions) {
  statements;
} else {
  statements;
}
```

Like with functions, each part of the `if - else if - else` block is contained within curly brackets (`{ }`). There can be zero or more `else if` blocks. The `else` block is optional.

#### Comparison Operators

Operator	Description
<code>==</code>	Equals
<code>!=</code>	Doesn't equal
<code>===</code>	Strictly equals
<code>!==</code>	Doesn't strictly equal
<code>&gt;</code>	Is greater than
<code>&lt;</code>	Is less than
<code>&gt;=</code>	Is greater than or equal to
<code>&lt;=</code>	Is less than or equal to

Note the difference between `==` (equals) and `===` (strictly equals). For two objects to be **strictly equal** they must be of the same value **and** the same type, whereas to be **equal** they must only have the same value. See the code sample below:

## Code Sample:

---

### ConditionalsAndLoops/Demos/StrictlyEquals.html

---- C O D E    O M I T T E D ----

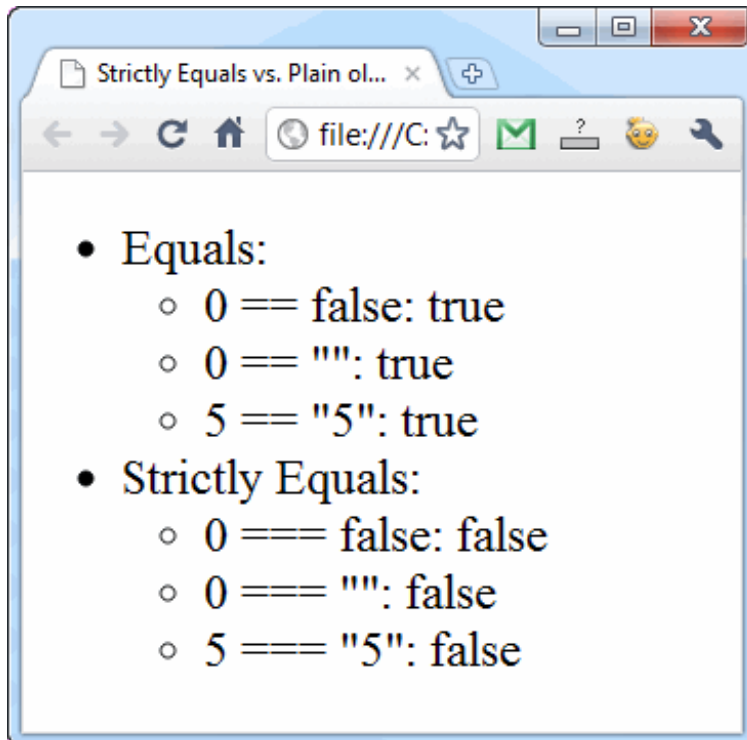
```
<li>Equals:
<ul>
  <li>0 == false:
    <script type="text/javascript">
      document.write(0 == false);
    </script>
  </li>
  <li>0 == "":
    <script type="text/javascript">
      document.write(0 == "");
    </script>
  </li>
  <li>5 == "5":
    <script type="text/javascript">
      document.write(5 == "5");
    </script>
  </li>
</ul>
</li>
<li>Strictly Equals:
<ul>
  <li>0 === false:
    <script type="text/javascript">
      document.write(0 === false);
    </script>
  </li>
  <li>0 === "":
    <script type="text/javascript">
      document.write(0 === "");
    </script>
  </li>
  <li>5 === "5":
    <script type="text/javascript">
      document.write(5 === "5");
```

```

</script>
</li>
</ul>
</li>
---- C O D E   O M I T T E D ----

```

Open this file in your browser to see that all of the value pairs are equal, but none is strictly equal:



### Logical Operators

Operator	Description
&&	and (a == b && c != d)
	or (a == b    c != d)
!	not ! (a == b    c != d)

The example below shows a function using an if - else if - else condition:

### Code Sample:

[ConditionalsAndLoops/Demos/IfElseifElse.html](#)

```

---- C O D E   O M I T T E D ----

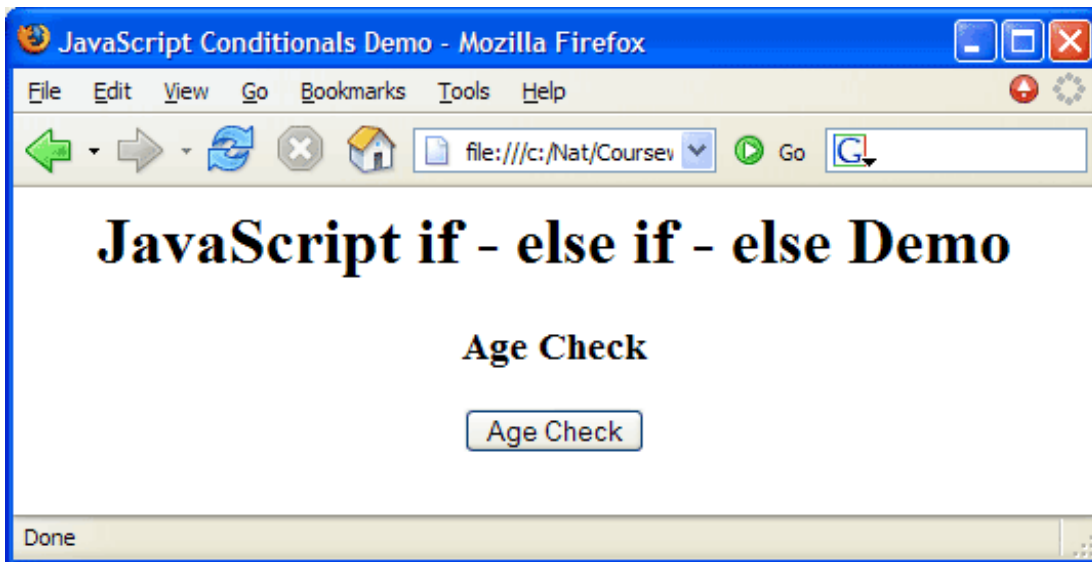
<script type="text/javascript">
function checkAge(){
    var age = prompt("Your age?", "") || "";

    if (age >= 21) {
        alert("You can vote and drink!");
    } else if (age >= 18) {
        alert("You can vote, but can't drink.");
    } else {
        alert("You cannot vote or drink.");
    }
}
</script>
---- C O D E   O M I T T E D ----

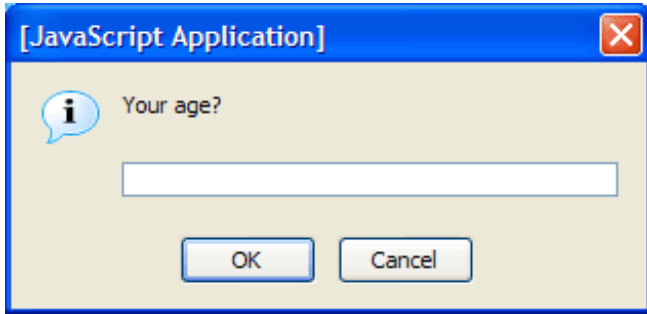
<form>
    <input type="button" value="Age Check" onclick="checkAge();">
</form>
---- C O D E   O M I T T E D ----

```

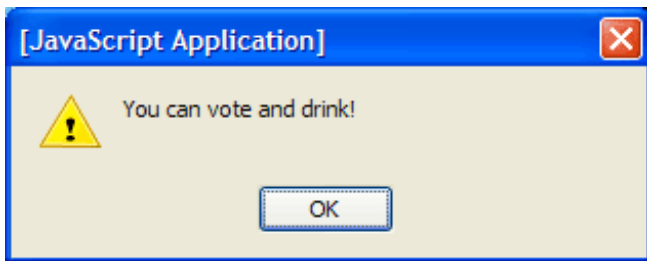
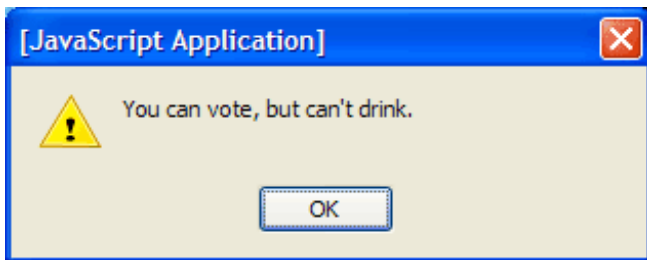
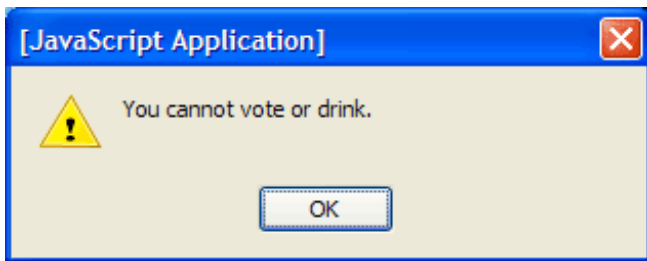
The display of the page is shown below:



When the user clicks on the **Age Check** button, the following prompt pops up:



After the user enters his age, an alert pops up. The text of the alert depends on the user's age. The three possibilities are shown below:



Also notice the use of the default operator (`||`) in the `checkAge()` function. The default operator was covered in the "JavaScript Operators" activity of the "Variables, Arrays and Operators" lesson. If you don't remember how it works, feel free to go back and review it.

## Compound Conditions

Compound conditions are conditions that check for multiple things. See the sample below.

```
if (age > 18 && isCitizen) {
    alert("You can vote!");
}

if (age >= 16 && (isCitizen || hasGreenCard)) {
    alert("You can work in the United States");
}
```

## Short-circuiting

JavaScript is lazy (or efficient) about processing compound conditions. As soon as it can determine the overall result of the compound condition, it stops looking at the remaining parts of the condition. This is useful for checking that a variable is of the right data type before you try to manipulate it.

To illustrate, take a look at the following sample:

### Code Sample:

---

#### ConditionalsAndLoops/Demos/PasswordCheckBroken.html

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Password Check</title>
<script type="text/javascript">
    var userPass = prompt("Password:", ""); //ESC here causes problems
    var password = "xyz";
</script>
</head>
<body>
<script type="text/javascript">
    if (userPass.toLowerCase() == password) {
        document.write("<h1>Welcome!</h1>");
    }
</script>
</body>
</html>
```

```

    } else {
        document.write("<h1>Bad Password!</h1>");
    }
</script>
</body>
</html>

```

Everything works fine as long as the user does what you expect. However, if the user clicks on the Cancel button when prompted for a password, the value `null` will be assigned to `userPass`. Because `null` is not a string, it does not have the `toLowerCase()` method. So the following line will result in a JavaScript error:

```
if (userPass.toLowerCase() == password)
```

This can be fixed by using the `typeof()` function to first check if `userPass` is a string as shown in the sample below.

## Code Sample:

---

### ConditionalsAndLoops/Demos/PasswordCheck.html

```

<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Password Check</title>
<script type="text/javascript">
    var userPass = prompt("Password:", "") || "";
    var password = "xyz";
</script>
</head>
<body>
<script type="text/javascript">
    if (typeof userPass == "string" && userPass.toLowerCase() == password) {
        document.write("<h1>Welcome!</h1>");
    } else {

```



```

    document.write("<h1>Bad Password!</h1>");
}
</script>
</body>
</html>

```

## Switch / Case Conditional Statements

The structure of a switch / case condition is shown below:

```

switch (expression) {
    case value :
        statements;
    case value :
        statements;
    default :
        statements;
}

```

Like if - else if - else statements, switch/case statements are used to run different code at different times. Generally, switch/case statements run faster than if - else if - else statements, but they are limited to checking for equality. Each case is checked to see if the *expression* matches the *value*.

Take a look at the following example:

### Code Sample:

#### ConditionalsAndLoops/Demos/SwitchWithoutBreak.html

```

<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Switch</title>
<script type="text/javascript">
    var quantity = 1;
    switch (quantity) {
        case 1 :

```

```

    alert("quantity is 1");
case 2 :
    alert("quantity is 2");
default :
    alert("quantity is not 1 or 2");
}
</script>
</head>
<body>
    <p>Nothing to show here.</p>
</body>
</html>

```

When you run this page in a browser, you'll see that all three alerts pop up, even though only the first case is a match. That's because if a match is found, none of the remaining cases is checked and all the remaining statements in the `switch` block are executed. To stop this process, you can insert a `break` statement, which will end the processing of the `switch` statement.

The corrected code is shown in the example below.

## Code Sample:

---

### ConditionalsAndLoops/Demos/SwitchWithBreak.html

```

<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Switch</title>
<script type="text/javascript">
    var quantity = 1;
    switch (quantity) {
        case 1 :
            alert("quantity is 1");
            break;
        case 2 :
            alert("quantity is 2");
            break;
        default :

```

```

    alert("quantity is not 1 or 2");
}
</script>
</head>
<body>
    <p>Nothing to show here.</p>
</body>
</html>

```

The following example shows how a `switch/case` statement can be used to record the user's browser type:

## **Code Sample:**

### **ConditionalsAndLoops/Demos/BrowserSniffer.html**

```

<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Simple Browser Sniffer</title>
<script type="text/javascript">
    switch (navigator.appName) {
        case "Microsoft Internet Explorer" :
            alert("This is IE!");
            break;
        case "Opera" :
            alert("This is Opera!");
            break;
        default :
            alert("This is something other than IE or Opera!");
    }
</script>
</head>
<body>
    <p>Your browser's <code>navigator.appName</code> is:
    <script type="text/javascript">
        document.write(navigator.appName);
    </script>.
    </p>
</body>
</html>

```

The `navigator` object, which is a child of `window`, holds

information about the user's browser. In this case we are looking at the `appName` property, which has a value of "Microsoft Internet Explorer" for Internet Explorer and "Netscape" for Mozilla-based browsers (e.g, Firefox, Chrome).

In conditional statements it's generally a good practice to test for the most likely cases/matches first so the browser can find the correct code to execute more quickly.

## Lesson 1, Activity 4: **Conditional Processing**

Duration: 20 to 30 minutes.

In this exercise, you will practice using conditional processing.

1. Open [ConditionalsAndLoops/Exercises/Conditionals.html](#) for editing.
2. Notice that there is an `onload` event handler that calls the `greetUser()` function. Create this function in the `script` block.
3. The function should do the following:
  1. Prompt the user for his/her gender and last name and store the results in variables.
  2. If the user enters a gender other than "Male" or "Female", prompt him/her to try again.
  3. If the user leaves the last name blank, prompt him/her to try again.
  4. If the user enters a number for the last name, tell him/her that a last name can't be a number and prompt him/her to try again.
  5. After collecting the gender and last name:
    - If the gender is valid, pop up an alert that greets the user appropriately (e.g, "Hello Ms. Smith!")
    - If the gender is not valid, pop up an alert that reads something like "XYZ is not a gender!"
4. Test your solution in a browser.

### **Challenge**

1. Allow the user to enter his/her gender in any case.
2. If the user enters a last name that does not start with a capital

letter, prompt him/her to try again.

## Solution:

---

### ConditionalsAndLoops/Solutions/Conditionals.html

---- C O D E    O M I T T E D ----

```
<script type="text/javascript">
function greetUser(){
    var gender, lastName;

    gender = prompt("Are you Male or Female?", "") || "";
    if (gender != "Male" && gender != "Female") {
        gender = prompt("Try again: Male or Female?", "") || "";
    }

    lastName = prompt("What's your last name?", "") || "";
    if (lastName.length === 0) {
        lastName = prompt("You must have a last name. Please re-enter:", "") || "";
    } else if (!isNaN(lastName)) {
        lastName = prompt("A last name can't be a number. Please re-enter:", "") || "";
    }

    switch (gender) {
    case "Male" :
        alert("Hello Mr. " + lastName + "!");
        break;
    case "Female" :
        alert("Hello Ms. " + lastName + "!");
        break;
    default :
        alert(gender + " is not a gender!");
    }
}
</script>
```

---- C O D E    O M I T T E D ----

## Challenge Solution:

---

### ConditionalsAndLoops/Solutions/Conditionals-challenge.html

---- C O D E    O M I T T E D ----

```
<script type="text/javascript">
```

```

function greetUser(){
  var gender, lastName;

  gender = prompt("Are you Male or Female?", ""); || "";
  if (gender.toLowerCase() != "male" && gender.toLowerCase() != "female") {
    gender = prompt("Try again: Male or Female?", "") || "";
  }

  lastName = prompt("What's your last name?", "");
  if (lastName.length === 0) {
    lastName = prompt("You must have a last name. Please re-enter:", "");
  } else if (!isNaN(lastName)) {
    lastName = prompt("A last name can't be a number. Please re-enter:", "");
  } else if (lastName.substr(0, 1) == lastName.substr(0, 1).toLowerCase()) {
    lastName = prompt("The first letter must be capitalized. Re-enter:", "") || "";
  }

  switch (gender.toLowerCase()) {
  case "male" :
    alert("Hello Mr. " + lastName + "!");
    break;
  case "female" :
    alert("Hello Ms. " + lastName + "!");
    break;
  default :
    alert(gender + " is not a gender!");
  }
}
</script>
---- C O D E   O M I T T E D ----

```

## Lesson 1, Activity 5: Loops

There are several types of loops in JavaScript.

- `while`
- `do...while`
- `for`
- `for...in`

### while Loop Syntax

```
while (conditions) {  
    statements;  
}
```

Something, usually a statement within the `while` block, must cause the condition to change so that it eventually becomes false and causes the loop to end. Otherwise, you get stuck in an infinite loop, which can bring down the browser.

### do...while Loop Syntax

```
do {  
    statements;  
} while (conditions);
```

Again something, usually a statement within the `do` block, must cause the condition to change so that it eventually becomes false and causes the loop to end.

Unlike with `while` loops, the statements in `do...while` loops will always execute at least one time because the conditions are not checked until the end of each iteration.



## for Loop Syntax

```
for (initialization; conditions; change) {  
    statements;  
}
```

In for loops, the initialization, conditions, and change are all placed up front and separated by semi-colons. This makes it easy to remember to include a change statement that will eventually cause the loop to end.

`for` loops are often used to iterate through arrays. The `length` property of an array can be used to check how many elements the array contains. We'll take a look at using the for loop in the upcoming presentation.

## for...in Loop Syntax

```
for (var item in array) {  
    statements;  
}
```

`for...in` loops are specifically designed for looping through arrays. For reasons that will be better understood when we look at object augmentation, the above syntax has a slight flaw. If the `Array` class is changed, it is possible that the `for...in` loop includes more items than what you anticipated.

To be on the safe side, we suggest that you use a more verbose syntax as seen below.

```
for (var item in array) if (array.hasOwnProperty(item)) {  
    statements;  
}
```

```
}

```

The `hasOwnProperty()` call will ensure that the item is indeed an element that you added to the array, not something that was inherited because of object augmentation.

## Code Sample:

---

### ConditionalsAndLoops/Demos/Loops.html

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>JavaScript Loops</title>
<script type="text/javascript">
  var index;
  var beatles = [];
  beatles["Guitar1"] = "John";
  beatles["Bass"] = "Paul";
  beatles["Guitar2"] = "George";
  beatles["Drums"] = "Ringo";
  var ramones = ["Joey", "Johnny", "Dee Dee", "Mark"];
</script>
</head>
<body>
<h1>JavaScript Loops</h1>
<h2>while Loop</h2>
<script type="text/javascript">
  index = 0;
  while (index < 5) {
    document.write(index);
    index++;
  }
</script>

<h2>do...while Loop</h2>
<script type="text/javascript">
  index = 1;
  do {
    document.write(index);
    index++;
  } while (index < 5);
</script>

<h2>for Loop</h2>
```

```

<script type="text/javascript">
for (index=0; index<=5; index++) {
  document.write(index);
}
</script>

<h2>for Loop with Arrays</h2>
<ol>
<script type="text/javascript">
var len = ramones.length;
for (var index=0; index<len; index++) {
  document.write("<li>" + ramones[index] + "</li>");
}
</script>
</ol>

<h2>for...in Loop</h2>
<ol>
<script type="text/javascript">
for (var prop in beatles) {
  document.write("<li>" + prop + " : " + beatles[prop] + "</li>");
}
</script>
</ol>
</body>
</html>

```

The sample above shows demos of all the aforementioned loops.

## Lesson 1, Activity 7: **Working with Loops**

Duration: 20 to 30 minutes.

In this exercise, you will practice working with loops.

1. Open [ConditionalsAndLoops/Exercises/Loops.html](#) for editing. You will see that this file is similar to the solution from the last exercise.
2. Declare an additional variable called `greeting`.
3. Create an array called `presidents` that contains the last names of four or more past presidents.
4. Currently, the user only gets two tries to enter a valid `gender` and `lastName`. Modify the code so that, in both cases, the user continues to get prompted until he enters valid data.
5. Change the `switch` block so that it assigns an appropriate value (e.g, "Hello Ms. Smith") to the `greeting` variable rather than popping up an alert.
6. After the `switch` block, write code that alerts the user by name if he has the same last name as a president. There is no need to alert those people who have non-presidential names.

### Challenge

1. Modify the code so that the first prompt for gender reads "What gender are you: Male or Female?", but all subsequent prompts for gender read "You must enter 'Male' or 'Female'. Try again:".
2. Modify the code so that the first prompt for last name reads "Enter your last name:", but all subsequent prompts for last name read "Please enter a valid last name:".
3. If the user presses the *Cancel* button on a prompt dialog, it returns `null`. Test this. It very likely results in a JavaScript error. If so, fix the code so that no JavaScript error occurs.

4. For those people who do not have presidential names, pop up an alert that tells them their names are not presidential.

## Solution:

---

### ConditionalsAndLoops/Solutions/Loops.html

```

---- C O D E   O M I T T E D ----

<script type="text/javascript">
function greetUser(){
  var gender, lastName, greeting;
  var presidents = ["Washington", "Jefferson", "Lincoln", "Kennedy"];

  do {
    gender = prompt("What gender are you: Male or Female?", "") || "";
  } while (gender.toLowerCase() != "male" && gender.toLowerCase() != "female");

  do {
    lastName = prompt("Enter your last name:", "") || "";
  } while (lastName.length === 0 || !isNaN(lastName));

  switch (gender.toLowerCase()) {
    case "male" :
      greeting = "Hello Mr. " + lastName + "!";
      break;
    case "female" :
      greeting = "Hello Ms. " + lastName + "!";
      break;
    default :
      greeting = "Hello Stranger!";
  }

  for (var pres in presidents) {
    if (lastName.toLowerCase() == presidents[pres].toLowerCase()) {
      alert(greeting + " You have the same last name as a president!");
    }
  }
}
</script>
---- C O D E   O M I T T E D ----

```

## Challenge Solution:

---

### ConditionalsAndLoops/Solutions/Loops-challenge.html

---- C O D E    O M I T T E D ----

```
<script type="text/javascript">
function greetUser(){
  var gender, lastName, greeting;
  var presidents = ["Washington", "Jefferson", "Lincoln", "Kennedy"];

  var repeat = false;

  do {
    if (!repeat) {
      gender = prompt("What gender are you: Male or Female?", "") || "";
      repeat = true;
    } else {
      gender = prompt("You must enter 'Male' or 'Female'. Try again:", "") || "";
    }
  } while (typeof gender != "string"
    || (gender.toLowerCase() != "male"
    && gender.toLowerCase() != "female"));

  repeat = false;

  do {
    if (!repeat) {
      lastName = prompt("Enter your last name:", "") || "";
      repeat=true;
    } else {
      lastName = prompt("Please enter a valid last name:", "") || "";
    }
  } while (typeof lastName != "string"
    || (lastName.length === 0
    || ! isNaN(lastName)));

  switch (gender.toLowerCase()) {
  case "male" :
    greeting = "Hello Mr. " + lastName + "!";
    break;
  case "female" :
    greeting = "Hello Ms. " + lastName + "!";
    break;
  default :
    greeting = "Hello Stranger!";
  }

  var match = false;

  for (var pres in presidents) if (presidents.hasOwnProperty(pres)) {
    if (lastName.toLowerCase() == presidents[pres].toLowerCase()) {
      alert(greeting + " You have the same last name as a president!");
    }
  }
}
```

```
    match = true;
  }
}

if (!match) {
  alert(greeting + " Your last name is not presidential.");
}
}
</script>
---- C O D E   O M I T T E D ----
```