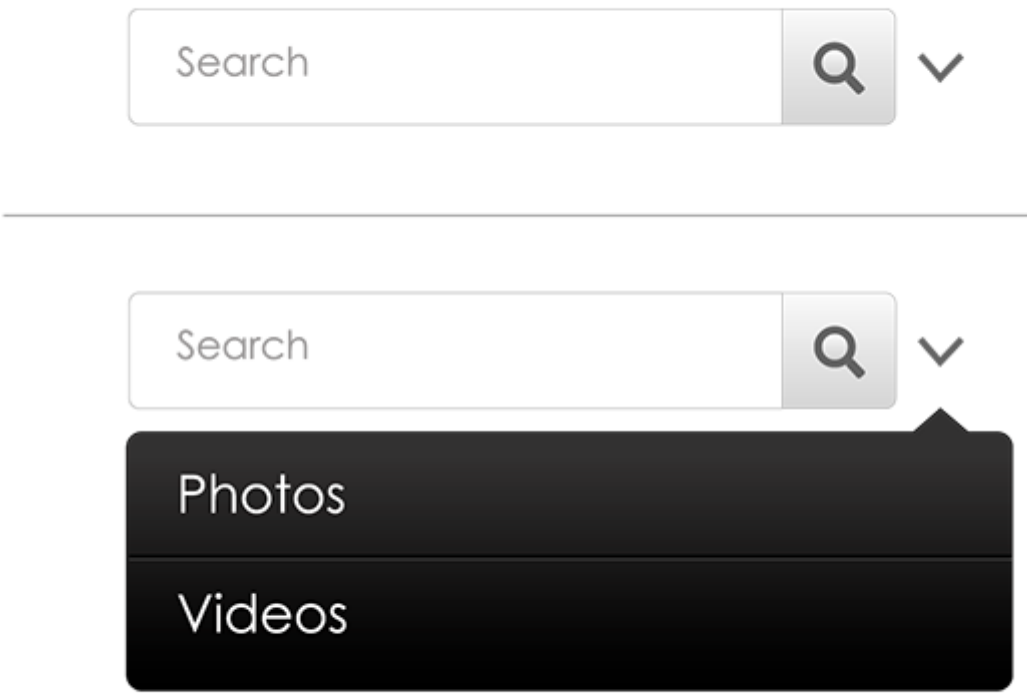
Introduction

Welcome Back!

In today's lesson, you'll get to use some of what you've learned so far in this course to create something really useful—a Search My Site function. Using JavaScript you can add a Search My Site capability to any website.



You'll also learn how to create drop-down list controls and use them for decision-making in your JavaScript code. Which can be used to add attributes to your search function!

We have a lot of ground to cover. So let's get right into it!

Search My Site and Drop Down

In today's lesson, you will be creating a search function that will allow you to search any domain using Google or Bing. This is a useful feature that can be added to any site.

Download

Use the link below to download Lesson 5 Class Project ZIP file. Inside the ZIP is a page you can take for a spin in any browser.

Search My Site: For starters, you'll need a new webpage with some HTML controls to allow users to type their search words and do the search.

You're welcome to type this page from scratch, if you're so inclined. Or, if you're not in the mood to practice typing right now, feel free to simply save them to your *Intro JavaScript* folder.

Download Lesson-05_ClassProject.zip

1 KB (Gigabytes) ZIP

Chapter 1: Coding Search My Site

Laying the Groundwork

In this chapter, we'll get started building the code for a Search My Site capability that you can add to any website you create.

Here are the Steps

1. Open *SearchMySite.html* in your browser, you'll see the content and controls.

   The controls don't do anything yet, because you need JavaScript code to actually perform the search. So let's get started on the JavaScript!

2. Open *SearchMySite.html* in your editor.

3. Make a space for a new line of code below the <title> and </title> tag but above </head>.

4. Place your cursor in the space you made and type a *<script>* tag.

5. Press **ENTER** and type a *</script>* tag.

Tip

Technically, it's not necessary to type both tags right off the bat. But typing both the opening and closing tags up front makes it impossible to forget to type the closing tag later, which is a very common error.

6. You may have noticed that *onclick="search()"* has already been added in the input tag for the button. So the code expects to find a JavaScript function named *search()* when the user clicks the button. Let's create that function now.

7. Put the cursor after the opening <script> tag, and press **ENTER** to add a new line for code inside the <script>. . . </script> tags.

8. Let's start with a JavaScript comment, by typing the following:

```
//Search a specific site rather than the entire Web
```

Tip

Though comments aren't required, they can be helpful later when reviewing code later.

9. Then press **ENTER**.

10. Type *function search(){}*.

Note

When you type *function search(){}* there should only be one space, the one between *function* and *search*. You don't want to arbitrarily add more spaces in there.

11. Then place your cursor between the curly braces and press **ENTER** a few times to add some blank lines.

Remember

Once again, we've had you type the closing curly brace for the function right away. That's not because you're required to type things that way. It's just to avoid the common mistake of forgetting to type it later.

Sneak Peek

Now you're <script> code should look like this:

```
<head>
 <title>Search My Site</title>
 <script>
 //Search a specific site rather than the entire Web
 function search(){


 }
 </script>
</head>
```

So now you have a JavaScript function named *search()* in the page. It doesn't do anything, because you haven't put any code inside that search function yet. The important thing to remember is that all code that the function executes must be placed between its opening and closing curly braces.

A Call to Action

Before we write the code to define what this function does, let's outline the steps that the code must perform when the function is called:

- ✓ Specify the domain name of the site to be searched (your website).
- ✓ Retrieve whatever the user typed into the search textbox.
- ✓ If the user added text in the textbox, create a URL to perform the search, and put it in the Address bar.
- ✓ If the user left the textbox blank, show an alert asking the user to type some text in the box.

Let's start by specifying the domain to search.

Specifying the Domain

In real life, this would be the domain name of your own site or whatever site you want to search. However, for now we're going to assume that you don't have a large site out on the public web. Instead, we're going to use Wikipedia.org.

Here are the Steps

1. Open *SearchMySite.html* in your editor.

2. Place your cursor on the empty line between the curly braces for the *search() function*.

3. As a note to yourself for future reference, type the following comment:

```
//Replace sample domain name below with your own domain name
```

4. Press **ENTER**, and then type:

```
var site = "wikipedia.org";
```

5. Press **ENTER** again to add a new blank line.

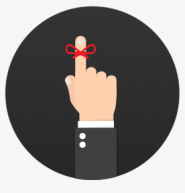
6. Then press **CTRL + S** to save your progress.

You can add any domain you like.

Feel free to replace wikipedia.org with your own domain name (if you have one) or the domain name of some other site to search (like Google).

Retrieving the Text Input

Next, the code needs to grab whatever text the user typed into the search box. The input tag for the textbox contains *id="txtlookfor"*, so we can use document.getElementById to find that textbox and get its value.



Remember

In JavaScript terminology, the *value* of a textbox is whatever text the user typed into the textbox.

Here are the Steps

1. Place your cursor on the blank line you created, right under *var site =* line you just typed.

Important

Make sure you're still inside curly braces. You don't want to go past the *}* that marks the end of the search function.

Type the following comment:

```
//Get the text that the user typed into the textbox
```

2. Press **ENTER**, and carefully type the code for your *lookfor* variable:

```
var lookfor = document.getElementById("txtlookfor").value;
```

3. Then press **ENTER** again and save your progress.

Sneak Peek

At this point your code should look like this:

```
<script>
 //Search a specific site rather than the entire Web
 function search(){
     //Replace sample domain name below with your own domain name
     var site = "wikipedia.org";
     //Get the text that the user typed into the textbox
     var lookfor = document.getElementById("txtlookfor").value;
}
 </script>
```

That last line of code you added creates a variable named *lookfor* which will store whatever text the user types into the **Search** textbox.

Remember

Keep in mind that JavaScript code is case-sensitive. This means that the JavaScript code you type has to exactly match the JavaScript code we're showing you, even the same uppercase and lowercase letters. Otherwise, your code might not work.

We're not quite done yet. We still have to write some code to perform the actual search. But we'll finish up this function in Chapter 2.

Chapter 2: Executing the Search

Adding Logic

Using if/else statements

In the last chapter, you wrote some HTML code to display a search box and button on a page. You wrote a function that, so far, identifies the site to search and grabs the text to search for out of the textbox. In this chapter, you'll write the code to perform the following actions:

- ✓ Determine if the user typed text into the search box.
- ✓ If yes, then the code will search for that text.
- ✓ If no, then the page will show an alert message asking the user to type some text to search for.

To do this we'll begin by putting in your basic *if . . . else* block.

Here are the Steps

1. Open *SearchMySite.html* in your editor now.
2. Place your cursor at the end of the *var lookfor =* line of code but before the curly brace that marks the close of the *search()* function.
3. Press **ENTER** and then type the following comment:

```
//If the box wasn't empty do the search
```

4. Then press **ENTER** again.
5. Now, type the basic skeleton of an *if . . . else* block:

```
if() {

}

else {

}
```

Remember

Again be sure to put in all the necessary opening and closing curly braces and parentheses so you don't forget to type them later. And don't forget to use all lowercase letters.

The *if* needs to make a decision based on whether or not the textbox was empty. How can we express that code?

Well, we already stored the contents of the textbox in the variable named *lookfor*. So whatever was in the textbox is now in that variable. It's data type is a string because anything that the user types into a textbox is defined as a string by default.

Fortunately for us, JavaScript includes a *.length* property that you can use with strings to determine their length. That length is the number of characters in the string.

- **If the user typed nothing into the box,** then the length of the *lookfor* string will be 0 (zero).
- **If the user typed anything into the box,** it will be some number *greater than zero* (>0).

So to determine if the *lookfor* variable contains any text, we just have to check

```
lookfor.length>0
```

for:

6. Put the cursor inside the parentheses of the *if() statement* where the condition goes, and type the following code:

```
lookfor.length>0
```

Important

Make sure you type the number zero, not the letter o.

7. Save your changes to the page.

If the length of *lookfor* is greater than zero, we want to do the search. Let's do that next.

Coding the Search

Creating the Search URL

To use Google to do the search, the code will need to send Google a URL that tells it the *searchwords* (words to search for) and the *domain* (where to search for them). The URL looks something like this:

```
https://www.google.com/search?q=searchwords site:
domain
```

The actual code will to substitute whatever the user typed in the textbox for *searchwords* and the website to be searched for *domain*. Thankfully, we have that information stored in variables named *lookfor* and *site*. So we just have to use a common programming practice called *string concatenation* to build the URL.

To concatenate strings in JavaScript, we use the JavaScript *+ operator*. When used with strings (rather than numbers), the + operator joins multiple strings into a single string.For instance *"Hello"+"World"* creates the string *"HelloWorld"*. So our script needs to use some + operators to string together a part that's always the same with the information from the variables in our script.

Let's give it a try!

Here are the Steps

1. Open *SearchMySite.html* in your editor.

2. Place your cursor between the curly braces for the *if (lookfor.length > 0) statement and press **ENTER**.*

3. *Then,* type the comment:

```
//Build URL for the search
```

4. Press **ENTER** again, and type the following code below the comment:

```
var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;
```

5. Save your progress.

Sneak Peek

That line of code is a doozy, and we'll dissect it in a moment. But first, check to make sure you're still on track in your own JavaScript code. It should look like this in your *SearchMySite.html* page:

```
<script>

//Search a specific site rather than the entire Web

function search(){

//Replace sample domain name below with your own domain name

var site = "wikipedia.org";

//Get the text that the user typed into the textbox

var lookfor = document.getElementById("txtlookfor").value;

//If the box wasn't empty do the search. Then press ENTER

if(lookfor.length>0) {

//Build URL for the search

var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;

}

else {

}

}

</script>
```

Let's look closely at the last line of code you just typed:

```
var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;
```

This kind of string concatenation is pretty common in JavaScript because it allows you to mix *literal* (unchanging) text that never varies with text that can vary, which is stored in *variables*.

In this example, we created a variable named *query*:

```
var query=
```

Into that variable we put the literal text (which is enclosed in quotation marks):

```
var query="http://www.google.com/search?q="
```

Then to that we tacked on whatever word or words happened to be stored in the *lookfor* variable (which came from the textbox) using the encodeURIComponent() function:

```
var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + "
```

After adding the massaged *lookfor* variable, the code adds the literal text *"site:"*. We need to tack this onto the end so Google will search only a single site rather than the entire Web:

```
var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:"
```

However, we need to provide the domain name of the site to search. We've already stored that in a variable named *site*. So, now all we have to do is tack that variable's value after the literal *"site:"* text:

```
var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;
```

Example

So, let's say the user types *blue bird* into the textbox. Currently, the site variable pointing to *Wikipedia.org*, so after the line of code that creates the query variable is executed, the query variable contains this:

```
http://www.google.com/search?q=blue%20bird site:wikipedia.org
```

This is what you need to send to Google to have it search the Wikipedia.org site for the phrase *blue bird*. The *%20* represents a space. As you can see, the encodeURIComponent() function automatically converted the space to %20.

Launching the Search

So at this point, we have the URL that we need to send to Google to have it do the search. We just have to write a little more code to put that URL into the Address bar of the browser.

Here are the Steps

1. Go back to *SearchMySite.html* in your editor.

2. Place your cursor on the line below the *var query we just defined,* but still above the closing curly brace for the *if statement*.

3. Type the following comment:

```
//Set Address bar equal to query
```

4. Then press **ENTER** and type the following JavaScript code on the new line:

```
location.href=query;
```

Note

The *location* object in JavaScript represents the Address bar and the *href* property is its http reference (basically, the URL that's showing in it). By setting the value of the property to the contents of the *query* variable, we change the URL to the contents of the variable, and that's what performs the search.

5. Save your work on *SearchMySite.html*.

6. Let's take it for a spin. Open *SearchMySite.html* in a browser (or **Refresh** or **Reload** in the browser).

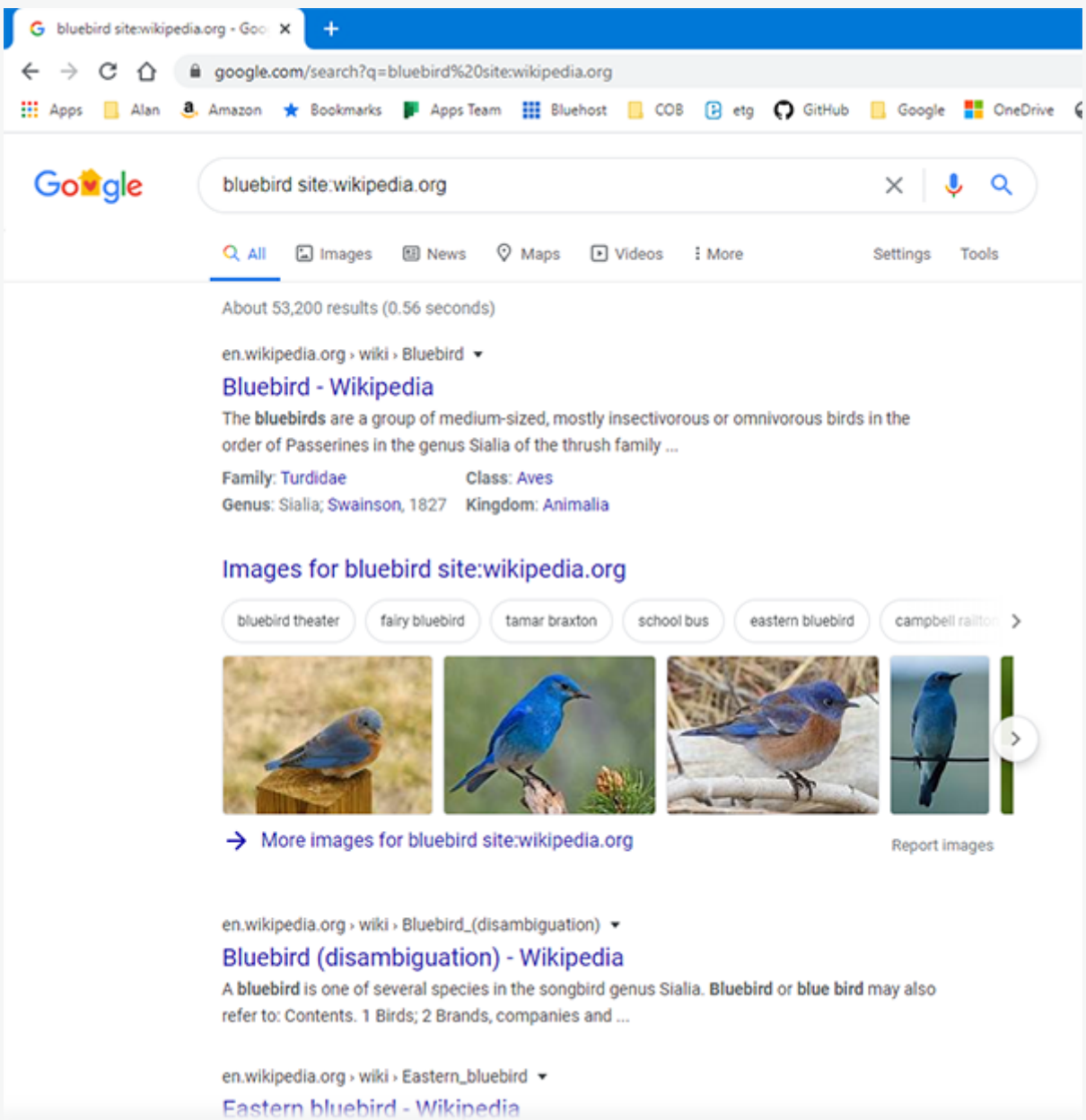
7. Now type the phrase *bluebird* (no space) into the search box and click the **Go** button.

Sneak Peek

If you've coded your page correctly, when you click Go, you should see the following URL appear in the address bar:

https://www.google.com/search?q=bluebird%20site:wikipedia.org

The page will display Google with the search results for the phrase *bluebird site:wikipedia.org*:



You might be thinking: "Big deal, we did a Google search." But it's not just any Google search. If you look through the search results, you'll see they're all from site you specified. So, when you use your own domain name in place of wikipedia.org in the code, the search results will be just from your site.

Customizing Your Search

You can type any word or phrase you like into the Search box:

- **If you use your own domain name in the script,** try to use words that actually appear on one of your pages.
- **If you leave the domain name as wikipedia.org,** you can type just about any words since it's such a huge site.

So far, so good. However, we're not completely done yet. Earlier, we put in an *if* statement that says *if* the user typed something into the search box, *then* do the search. We haven't yet told the code what to do if the user didn't type some text into the search box. Let's do that next.

What if there are no terms to search?

Adding an Alert

As we discussed earlier, when the user leaves the search box empty the variable *lookfor* has a length of zero. To let the code know how to deal with that situation, we'll need to use our *else* portion of the *if* statement, which we've already added to our code. So let's just fill in that part with an alert message.

Here are the Steps

1. Open *SearchMySite.html* in your editor.
2. Move the cursor to just inside the curly braces for the *else statement* and press **ENTER**.

Important

Make sure you're still above the two closing curly braces that mark the end of the *else statement* and the end of the *search* function.

3. On the line you created, type the following comment:

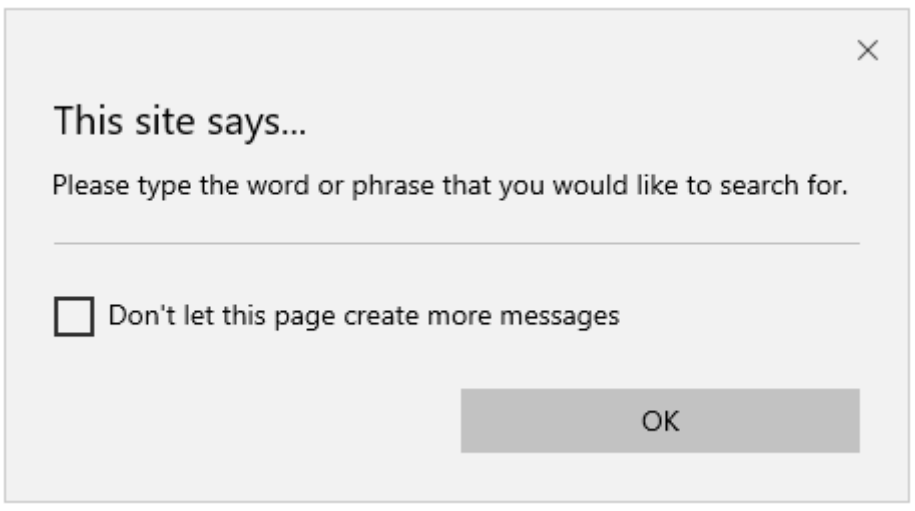
```
//If textbox was empty, show an alert
```

4. Then press **ENTER**.

5. Now add an alert that says *Please type the word or phrase that you would like to search for*.

```
alert("Please type the word or phrase that you would like to search for.");
```

- Now save your work.
- Let's give it a whirl! Open *SearchMySite.html* in a browser (or **Refresh** and **Reload**, if it's already open).
- This time, leave the search textbox empty, and click the **Go** button. You should see a warning that matches the alert you just coded (the exact appearance of the alert box depends on your browser):



- Click **OK** to close the warning.

Any user who was quick to click the **Go** button without typing some search word or phrase should now have a little better idea of how to use the search feature.

Sneak Peek

So now your entire page should look like the code block below.

```
<!DOCTYPE html>

<html>

<head>

<title>Search My Site</title>

<script>

//Search a specific site rather than the entire Web

function search(){

//Replace sample domain name below with your own domain name

var site = "wikipedia.org";

//Get the text that the user typed into the textbox

var lookfor = document.getElementById("txtlookfor").value;

//If the box wasn't empty do the search. Then press ENTER

if(lookfor.length>0) {

//Build URL for the search

var query="http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;

//Set Address bar equal to query

location.href=query;

}

else {

//If textbox was empty, show an alert

alert("Please type the word or phrase that you would like to search for.");

}

}
```

```
    </script>

    </head>

    <body>

    <p>Search this site:

    <input type="text" id="txtlookfor" autofocus>

    <input type="button" onclick="search()" value="Go">

    </p>

    </body>

    </html>
```

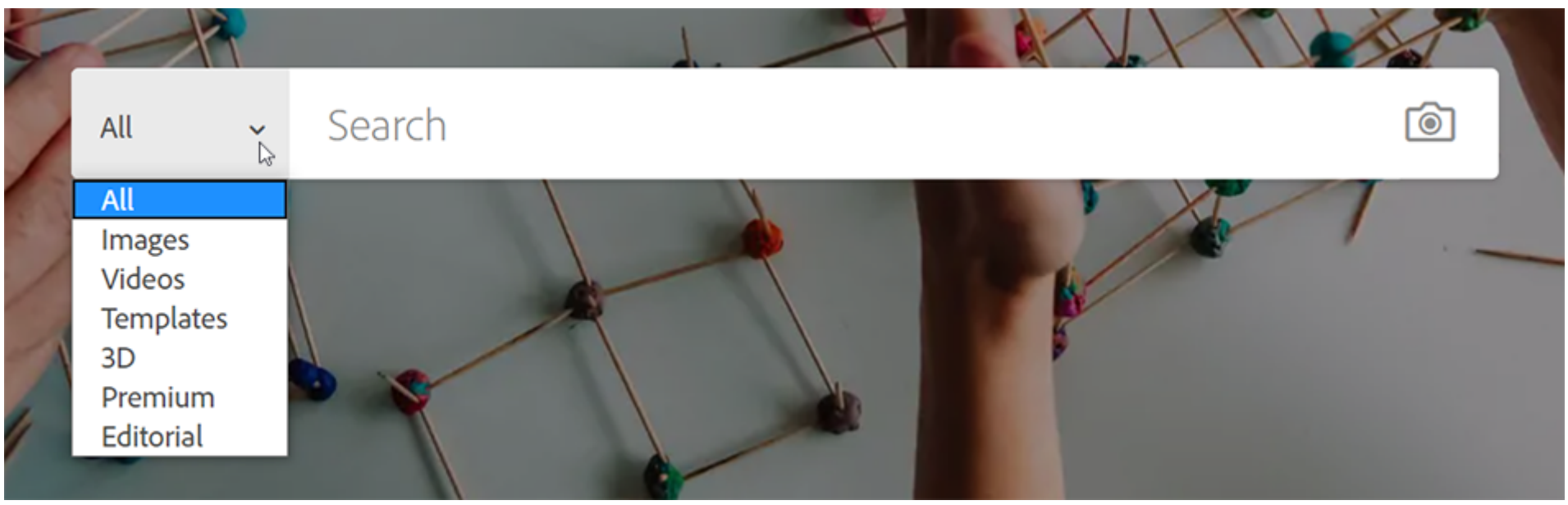
Right now, you basically have a free Search My Site capability that you can add to your own website, with just a relatively small JavaScript function. While our script currently uses Google to search the specified site, you can kick it up a notch by adding a drop-down list that lets the user choose between Bing or Google as the search engine. We'll show you how next.

Chapter 3: Selecting Your Search Engine

Adding a Drop-Down List

So far, you've learned how to create a Search My Site textbox and button that can be used in any site. It uses Google to search the site. Now, you'll add a drop-down list that lets the user select a search engine to use.

Drop-down lists are *controls* commonly found in many websites and other applications. You've probably seen a lot of them. The control looks like a textbox with an arrow on the side. When you click the arrow, a menu drops down so you can make a selection from a list of choices.



To add a drop-down list to a page, use the HTML <select> . . . </select> tags and <option> . . . </option> tags. Place the <select> tag where you want the control to appear on the page. Add *id="name"* to that tag. That's important because if you want to be able to get the user's selection from the list via JavaScript, the control needs an ID name. And that, in turn, is because JavaScript can only "find" controls that have ID names.

After the <select> tag, you can define each drop-down list item using <option> . . . </option> tags. Inside each <option> tag you can use *value=* followed by a number to give each item a unique number. JavaScript will determine the *value* of the control (in other words, it will determine which option the user picked) by looking at the value of the selected item.

At the end of the list, use </select> to mark the end of the items and the end of the control.

In the process, you'll learn how to create HTML drop-down list controls and have a chance to practice your JavaScript coding, too. Let's get right to it!

Let's give it a try by making a simple drop-down list with two options, Bing and Google, which you should recognize as common search engine names. We should also tell users what they're selecting from the box. So we'll put a little text in front of the control.

Here are the Steps

1. Open *SearchMySite.html* in your editor.

2. Move the cursor so it's at the end of the HTML tag *<input type="text" id="txtlookfor" autofocus>*, and press **ENTER**. We're going to put the select box between the textbox and the **Go** button.

3. Type the HTML comment *<!-- Show drop-down list -->*, and then press **ENTER**.

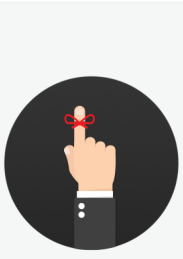
4. To add a nonbreaking space and a little text type the following text:

```
 Choose engine:
```

5. Then press **ENTER**.

6. Now, you need the code to show the drop-down list. Let's start by adding the *<select>* and *</select>* control tags under the * Choose engine:* line.

7. Place your cursor in between the <select> and </select> tags and press **ENTER** a couple of times to make some room for more code.



Remember

We put the closing tag in right after the opening tag. You're not required to type tags in a specific order but this practice helps to avoid the common error of forgetting to type the closing tag later.

8. Now we need to add an ID to the <select> control so that we can later use JavaScript to get the user's selection. So place your cursor after inside the <select> control tag, between the t and the > and type

id="dropdown" so that it looks something like this:

```
<select id="dropdown">
```

Adding IDs

Just like variable names, the ID name is just a name you make up. Meaningful names always make coding easier, so we used the name *dropdown*. It must start with a letter, cannot contain spaces, and is case-sensitive.

9. Next, you need to define the items that appear in the drop-down list. To do this you're going to add a pair of <option> and </option> tags. So, place your cursor after the *<select id="dropdown">* tag and press **ENTER**.

10. For your first option, type *<option value="1">Bing</option>* and press **ENTER**. This defines your first option as "Bing".

11. For your second option, type *<option value="2">Google</option>* and press **ENTER**. This defines your second option as "Google".

Defining Your Options

You may have noticed, that we gave each option a unique value using the *value=* attribute and a number. We can use that unique number to have JavaScript make decisions about what to do next based on the user's selection in the drop-down.

12. Save the page in your editor. Your code should look something like this:

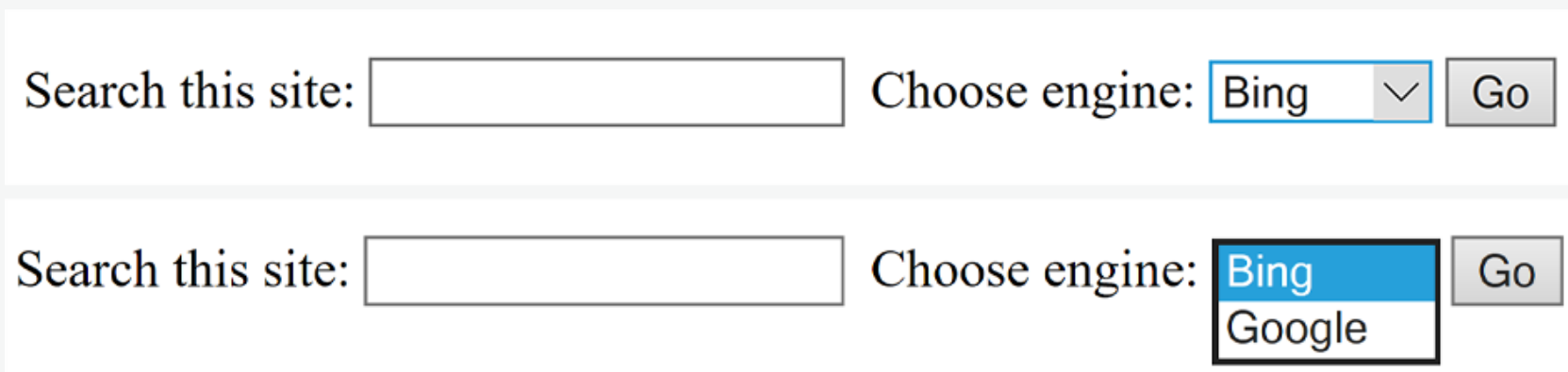
```
<body>
    <p>Search this site:
        <input type="text" id="txtlookfor" autofocus>
               <!-- Show drop-down list -->
                Choose engine:
               <select id="dropdown">
               <option value="1">Bing</option>
               <option value="2">Google</option>

               </select>
        <input type="button" onclick="search()" value="Go">
    </p>
</body>
```

13. Now, let's verify that you typed the HTML code correctly and everything is working as it should. Open *SearchMySite.html* in a browser, or reload or refresh the page if it's already in the browser.

Sneak Peak

If your code is right, you should see the new text "Choose engine:" followed by a drop-down list to the right of the textbox. When you click the arrow on the drop-down list, you should see Bing and Google in the drop-down list.



The code won't actually search Bing yet, because we need to add JavaScript code for that. But there's no point in working on the JavaScript code until you're sure the drop-down list is in place and is working properly on the page.

Changing the Default Option Selection

You may notice that when you first open the page in a browser, Bing shows as the selected engine in the drop-down list control. That's because a drop-down list will always show the first item in the list as the default selection. If you have a preference for which option you want to show as the default, there are two ways you can do this:

1. **You can move the option to the top of the list.** This means you move the whole <option> . . . </option> phrase so that it is the first item listed in the code.

2. **You can keep the current order and add the selected attribute to the desired option.** This means you add a space and the word *selected* to the <option> tag that you want to set as the default.

Ordering Your Drop-Down

There's never a *required* order for the options in a drop-down list, but there may be instances where a specific order is desired. Expected order may be more intuitive for some option lists than others, such as alphabetical order for words or smallest-to-largest for numbers. This is important to keep in mind when you are working with larger option lists.

Since we have a relatively short list, we'd like to make Google the default selection in our sample box without changing the order of items in the drop-down list.

Here are the Steps

1. Open *SearchMySite.html* in your editor.

2. Go to the <option value="2">Google</option> line of code.

3. Place your cursor just inside the > in the option open tag, after *value="2"* attribute.

4. Press **SPACE** and type the word *selected* (all lowercase). Now that line of code should look like this:

```
<option value="2" selected>Google</option>
```

5. Save the page in your editor.

6. Now open, reload or refresh the page in the browser.

Sneak Peek

If you typed all the code correctly, the initial selection in the drop-down list should now be Google rather than Bing. Of course, you can still click the drop-down arrow and change that if you like.

So far, we've added a drop-down list control to the page that allows the user to choose a search engine. However, JavaScript knows nothing about it because we haven't done anything in the JavaScript code to make the drop-down part of the process. To make the code "smarter," we need it to build one query string if the user chose Bing as the search engine and a different query string if the user chose Google as the search engine. Let's do that next!

Searching using the Chosen Engine

In order to tell the code what we want it to do, we need the following:

- ✓ A line of code to determine what the user chose in the drop-down list control.
- ✓ An *if . . . else* block that decides what query string to build based on that selection.

Let's do that now.

Determining the Drop-Down Choice

We need to create a line of code that grabs the user's choice in the drop-down menu. To do that, we need to create a variable using *document.getElementByID("dropdown ")*to identify the control.

Important

Be careful about your uppercase and lowercase letters, because it's case-sensitive.

Then we need to link the choice the user actually selects, so we'll add the *.value* property on the end of this line of code. This is the value that was assigned to the option (or selected item):

- If the user chooses Bing from the drop-down list (which has value="1" in its <option> tag), the value stored will be *1*.

- If the user chose Google from the drop-down list (which has <option value="2"> as its option tag), the value stored in that variable is *2*.

To make a long story short, the variable will contain *1* or *2* depending on the selection made by the user. We can then use that information to make another decision within the JavaScript code.

Here are the Steps

1. Open *SearchMySite.html* in your editor.

2. Move the cursor to the line of JavaScript code that reads if *(lookfor.length > 0) {* and press **ENTER** a couple of times to make some room.

3. Type the following comment:

```
//Look at the drop-down value and make search URL accordingly
```

4. Then press **ENTER** to move to a new line.

5. Type *var ddchoice = document.getElementById("dropdown").value* and press **ENTER**.

6. Save your progress.

Remember

The comment you just added, like all comments, is optional and is just there as a note to yourself about what the next line of code is doing.

The second line creates a variable named *ddchoice*. Like all variable names, *ddchoice* is just a name you make up. We used *ddchoice* because in it we're going to store the user's choice from the drop-down list in that variable (in other words, we're

Choice-based Search URLs

The next step is to write an appropriate search URL depending on the value of *ddchoice*. Right now, we just have a single line of code that starts with *var query = . . .* and then creates a string variable for searching Google. But that's not going to work anymore, because the user may want to search Bing rather than Google. So we need a block of JavaScript code that flows like this:

```
If the user chose Bing

Make a query string for Google

Else

Make a query string for Bing

End if
```

That's the correct logic. But we can't write it in the code that way, because it's not proper JavaScript syntax. Let's add the sample logic to our page using proper JavaScript syntax.

Here are the Steps

1. Go to *SearchMySite.html* in your editor.

2. Place your cursor at the end of the JavaScript comment *//Build URL for the search* and press **ENTER**.

3. Add the following if statement:

```
if (ddchoice == 2) {
```

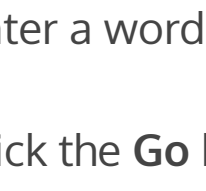
4. Then move your cursor to the end of the line of code that reads:

```
var query = "http://www.google.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;
```

5. Press **ENTER** and add the following block of code:

```
} else {
var query = "http://www.bing.com/search?q=" + encodeURIComponent(lookfor) + " site:" + site;
}
```

6. Save your progress.



Note

Notice the logic of the code. The *if* statement says that if the *ddchoice* variable contains a *2* (which means that the user chose Google from the drop-down list), then the variable named *query* should contain a URL that points to www.google.com (and it does). Else (if the user didn't choose *2*), the only other choice is Bing, so make search URL point to www.bing.com (which it does).

The rest of the code is the same as before. The line that reads *location.href = query;* sets the Address bar of the browser to whatever is in the query variable and performs the search.

7. Go ahead and take it for a spin! Open the *SearchMySite.html* page in your browser (or reload or refresh it in the browser).

8. Enter a word or phrase to search for.

9. Click the **Go** button, and it should search Google for pages in Wikipedia.org (or whatever domain name you provided) for that word or phrase.

10. Now, click the **Back** button to return to the search page.

11. Type the same word or phrase in the search box, but this time change the drop-down to **Bing**.

12. Then click **Go**, and it will search Bing for that same word or phrase.

Now users get to search your site using either Bing or Google (in case they have a preference). Hopefully, that went well for you. If not, check your code below.

Download

To check your code, you can download the correct code here:

Download Lesson-05_ClassProjectFinal.zip

1.6 KB (Gigabytes) ZIP

Join the Discussion

If you have any questions or concerns, be sure to stop by the Discussion Area and share your thoughts.

In the Assignment for this lesson, we'll show you how to work with checkboxes and radio buttons using JavaScript. For now, let's review all that you've learned in today's lesson.

Review

Whew, we covered a lot of ground in today's lesson. Some things you learned in previous lessons you were able to put to practical use in today's lesson and you also learned some new tricks. Let's review:

- **Coding Search My Site:** Here you laid the basic groundwork by creating your search function and adding a variable specifying the domain name of the site to be searched (your website). You also added a variable to retrieve whatever the user typed into the search textbox.

- **Executing the Search:** In this section you added some logic so that if the user adds text in the textbox, a URL is generated to perform the search. Otherwise, if the textbox is empty an alert appears to remind the user to type something in the textbox. You used JavaScript to take data from a textbox and a variable and combine it into a single string of text that can be used as the URL for performing such a search. You now know to concatenate (join) strings together using the JavaScript + operator.

- **Selecting Your Search Engine:** We added an option for the user to select either Google or Bing when performing the search. You learned how to make and use drop-down lists. To display a drop-down list control on a page, use the HTML <select> and <option> tags. You can use *selected* in any <option> tag of a drop-down list to make that option the default selection for the drop-down.

In the next lesson, we'll have some fun with JavaScript, sound effects, background music, and custom player controls. Along the way, you'll learn some tricks for detecting browser capabilities and applying CSS with JavaScript. Good stuff to know. See you there!

Lesson 5 Assignment

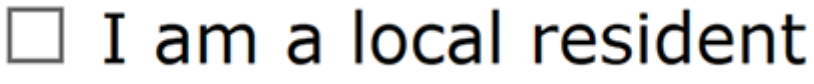
Radio Buttons and Check Boxes

What are Radio Buttons and Check Boxes?

In this lesson you saw how to work with a dropdown list, which you define with a <select> tag in HTML. Two other popular form controls are checkboxes and radio buttons:

- **Radio buttons:** A radio button allows you to choose one (and only one) of two or more mutually exclusive options. For example, the user can pick one, and only one, age group from the options below. Again, this is just a picture of a radio button group, so clicking the image below won't have any effect.

- **Checkbox:** A checkbox allows only for two options, either the box is checked, or it's not. You've probably seen a million of them.



Coding Radio Buttons and Check Boxes

Use the HTML input tag with type="radio" for any radio button. Give it a name, a unique value, and a unique id. Then to define the text to the right of the radio button, use a <label> tag with for= set to the id of the radio button. Here is sample HTML code to display the radio button list above:

```
<form id="frmAge">

<input type="radio" name="rbAge" value="0" id="age0">

<label for="age0">Under 12 years  </label>

<input type="radio" name="rbAge" value="10.00" id="age1">

<label for="age1">Adult  </label>

<input type="radio" name="rbAge" value="5.00" id="age2">

<label for="age2">65 or older</label>

</form>
```

Let's break it down:

Text equivalent start.

Topic	Information
`<form id="frmAge">`	Here we create a form that will house the radio inputs and assign it the id *frmAge*.
`<input type="radio" name="rbAge" value="0" id="age0">`	Here we added radio input the name, *rbAge*, to the radio input. This is just a name we made up. We also added a value of 0 and the id *age0*. Again this value and id is arbitrary, but you want to select something that make sense—since you may want to refer back to it in future code.
`<label for="age0">Under 12 years  </label>`	Note how the input id, *age0*, is used in the for= attribute of the label tag as well. This associates the label with the radio button. This allows the user to select the radio button by clicking the checkbox itself, or by clicking the text to the right.
`<input type="radio" name="rbAge" value="10.00" id="age1">`	This is next radio button that will be displayed. You'll notice that the radio input name is the same as the previous one: *rbAge*. The differences come in with the value, *10.00*, and the id, *age1*. This is because when you code, you can have it pull from any of the radio buttons named *rbAge*—but you want to specific the response using the value and id.
`<label for="age1">Adult  </label>`	The input id is used to link the label to the radio button so the user can click on the label to select the button.

Topic	Information
`<input type="radio" name="rbAge" value="5.00" id="age2">`	Here a third radio button is created with a value of *5.00* and an id of *age2*.
`<label for="age2">65 or older</label>`	Again, the input id is used to link the label to the radio button so the user can click on the label to select the button.
`</form>`	This denotes the end of the form.

Read the topic in the first column. Then read the second column for the information.

Text equivalent stop.

Name

Each <input type="radio"...> tag contains a name= attribute. That name is *not* the same as an ID, so it doesn't need to be unique to each item. Instead, the name is a way of saying all three radio buttons are part of one group (named rbAge) of mutually exclusive options, meaning only one option at a time can be selected. Selecting a radio button automatically unselects any radio button in the group that was already selected.

ID

Each radio button also have a unique id, specified with the usual id= attribute. That id allows label text to be associated with one button, where the label's for= attribute is the same as the id name. For example the tag <label for="age0"> defines the text that appears to the right of the radio button that has the id age0. The association allows the user to select an option either by clicking the actual radio button, or the text to the right of that button.

Value

Each radio button has a value, as identified by its value= attribute. That value is what we're generally interested in, from a JavaScript perspective, when looking to see which button the user selected.

- If the middle radio button is selected, the value returned will be 10.00.
- If none of the items in the radio button group is selected, the value retuned to JavaScript is an empty string, signified by a pair of single or double quotation marks with nothing in between, like this "".

At first glance this code looks complete, but its really just three pairs of <input> and <label> tags to define the radio button and text for each item. However, encasing the entire radio button group in <form> tags makes it easy to detect which radio button is selected within a group. This way you can just get the value for the form by it's id. For example, the value of the radio button group above can be determined using the following JavaScript code:

```
document.getElementById("frmAge").elements['rbAge'].value;
```

- That *frmAge* name is the id assigned to the form as a whole.
- The *rbAge* name refers to the group of radio buttons that share the name *rbAge*, as specified in each buttons name= attribute.
- The *value* that JavaScript returns is the value of the selected button, as indicated by that button's value= attribute.

As you can see the code follows a clear pattern, once you understand it.

Use the HTML input tag with type="checkbox" for any checkbox. Give it a unique id. Then to define the text to the right of the checkbox, use a <label> tag with for= set to the id of the checkbox. Here is sample HTML code that can be used to create the checkbox above:

```
<input type="checkbox" id="cbResident">
```

Let's break it down:

Text equivalent start.

Topic	Information
`<input type="checkbox" id="cbResident">`	Here we added the id name, cbResident, to the checkbox input. This is just a name we made up.
`<label for="cbResident">I am a local resident</label>`	Note how the id name, cbResident, is used in the for= attribute of the label tag as well. This associates the label with the checkbox. This allows the user to check or uncheck the checkbox by clicking the checkbox itself, or by clicking the text to the right.

Read the topic in the first column. Then read the second column for the information.

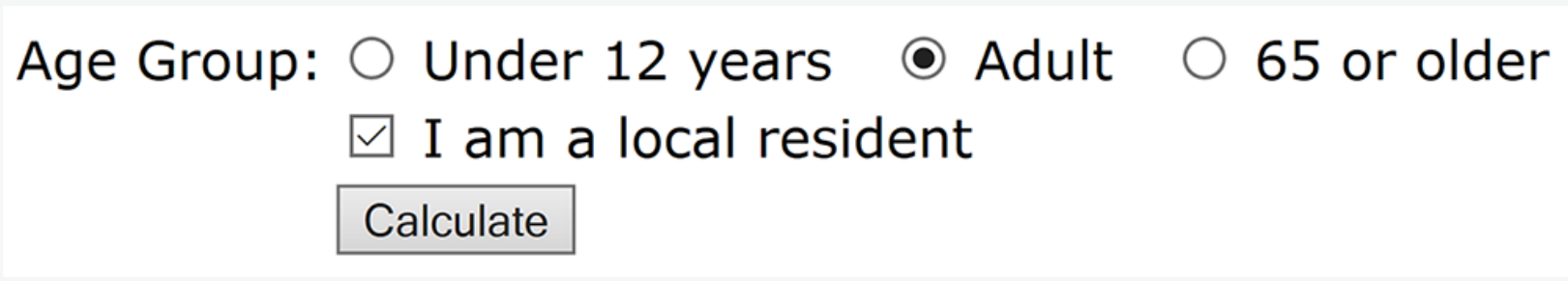
Text equivalent stop.

In JavaScript use the syntax *object*.checked to determine if a checkbox is checked. For example, the checkbox above is named cbResident. This line of JavaScript code returns true of that checkbox is checked, and returns false if the checkbox isn't checked:

```
document.getElementById("cbResident").checked
```

Download

For your assignment, we've constructed a web page that shows both of the form controls we reviewed above on a web page:



Use the link below to download the ZIP file which contains the following files that you'll need:

1. RadioButtonsandCheckBoxes.html
2. RadioButtonsandCheckBoxes.txt

L05-Assignment.zip

2 KB (Gigabytes) ZIP

You can retype or copy and paste if you'd like to run through the code, or you can simply download and save the files to your *Intro to JavaScript* folder.

Right now, you have a set of radio button inputs, a check box input, and a Calculate button—but they don't do anything yet. Here's what you're going to try and code:

- When the user clicks Calculate, the page should determine the price of admission based on the value of the selected radio button.
- However, if the checkbox is checked, they should get an additional $1.00 discount.
- The price for children under 12 is free (value=0).

- The price for an adult is $10.00 (value=10.00).
- The price for a senior is $5.00 (value=5.00).
- The result you can display in the <td id="result"></td> table cell.

Now, we know that is a significant challenge this early in the JavaScript learning curve. So we've included a video to walk you through it.

Download

To check your code, you can download the correct code here:

Download L05-Answers.zip

2.3 KB (Gigabytes) ZIP

Lesson 5 Resources for Further Learning

Adding Search Functionality to Your Website (https://www.lifewire.com/searching-your-site-3466200)

https://www.lifewire.com/searching-your-site-3466200

There are other ways you can add Search My Site to your website, without JavaScript. Some are free; some require that you pay a fee. You can find a review of some popular services on this site.

How Can I Make a Google Internal Site Search Script by JavaScript (https://stackoverflow.com/questions/3638753/how-can-i-make-a-google-internal-site-search-script-by-javascript)

https://stackoverflow.com/questions/3638753/how-can-i-make-a-google-internal-site-search-script-by-javascript

Here's the basic JavaScript code for the Google site search.

Strings (https://www.quirksmode.org/js/strings.html)

https://www.quirksmode.org/js/strings.html

Here's a tutorial on JavaScript strings, including a discussion of string concatenation.

JavaScript String Object (https://www.w3schools.com/jsref/jsref_obj_string.asp)

https://www.w3schools.com/jsref/jsref_obj_string.asp

Follow this link for a reference on JavaScript strings.

JavaScript Length Property (https://www.w3schools.com/jsref/jsref_length_string.asp)

https://www.w3schools.com/jsref/jsref_length_string.asp

Click this link for some brief documentation on string length.

---

encodeURIComponent Function (JavaScript) (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/encodeuricomponent)

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/encodeuricomponent

Here's some documentation for the JavaScript encodeURIComponent function.

HTML < select > Tag (https://www.w3schools.com/tags/tag_select.asp)

https://www.w3schools.com/tags/tag_select.asp

Click this link for a reference to the HTML <select> . . . </select> tags and drop-down list controls.

HTML DOM Select Object (https://www.w3schools.com/jsref/dom_obj_select.asp)

https://www.w3schools.com/jsref/dom_obj_select.asp

Here's a reference to drop-down list controls and ways you can access them from JavaScript.

Google Custom Search Engine (https://alansimpson.me/javascript/code_quickies/sitesearch/)

https://alansimpson.me/javascript/code_quickies/sitesearch/

Here's another way to add Search My Site to your site, using Google's JavaScript code.

Input Textbox Checked Property (https://www.w3schools.com/jsref/prop_checkbox_checked.asp)

https://www.w3schools.com/jsref/prop_checkbox_checked.asp

Here's is a page on using JavaScript with HTML checkboxes.

How to get value of selected radio button? (https://stackoverflow.com/questions/15839169/how-to-get-value-of-selected-radio-button)

https://stackoverflow.com/questions/15839169/how-to-get-value-of-selected-radio-button

Here is some discussion of getting the value of a selected radio button with JavaScript.