

More on Working with Forms

Lesson Objectives

When you complete this skills ladder lesson, you will be able to:

- use form collections.
- use the `elements` collection to access the value of a selected radio button.
- submit a form with JavaScript.

In this lesson, we temporarily leave our movie application to explore the `forms` collection. But don't worry, we'll come back to movies in the next lesson!

Using Form Collections

The document object contains an object named `forms` that contains all the forms in your page. This is known as the *forms collection*. So, instead of accessing your form like this:

```
var theForm = document.getElementById("theForm");
```

You can access your form like this:

```
var theForm = document.forms.theForm;
```

Let's try using the `forms` collection now. Take a moment to review the HTML as you're typing it in and notice the different kinds of form inputs we're using: text, checkbox, textarea, date, and of course a button to submit the values.

CODE TO TYPE:

```
<!doctype html>
<html lang="en">
<head>
  <title> Personality Quiz </title>
  <meta charset="utf-8">
  <script>
    window.onload = init;

    function init() {
      var submitButton = document.getElementById("submitButton");
      submitButton.onclick = getData;
```

```

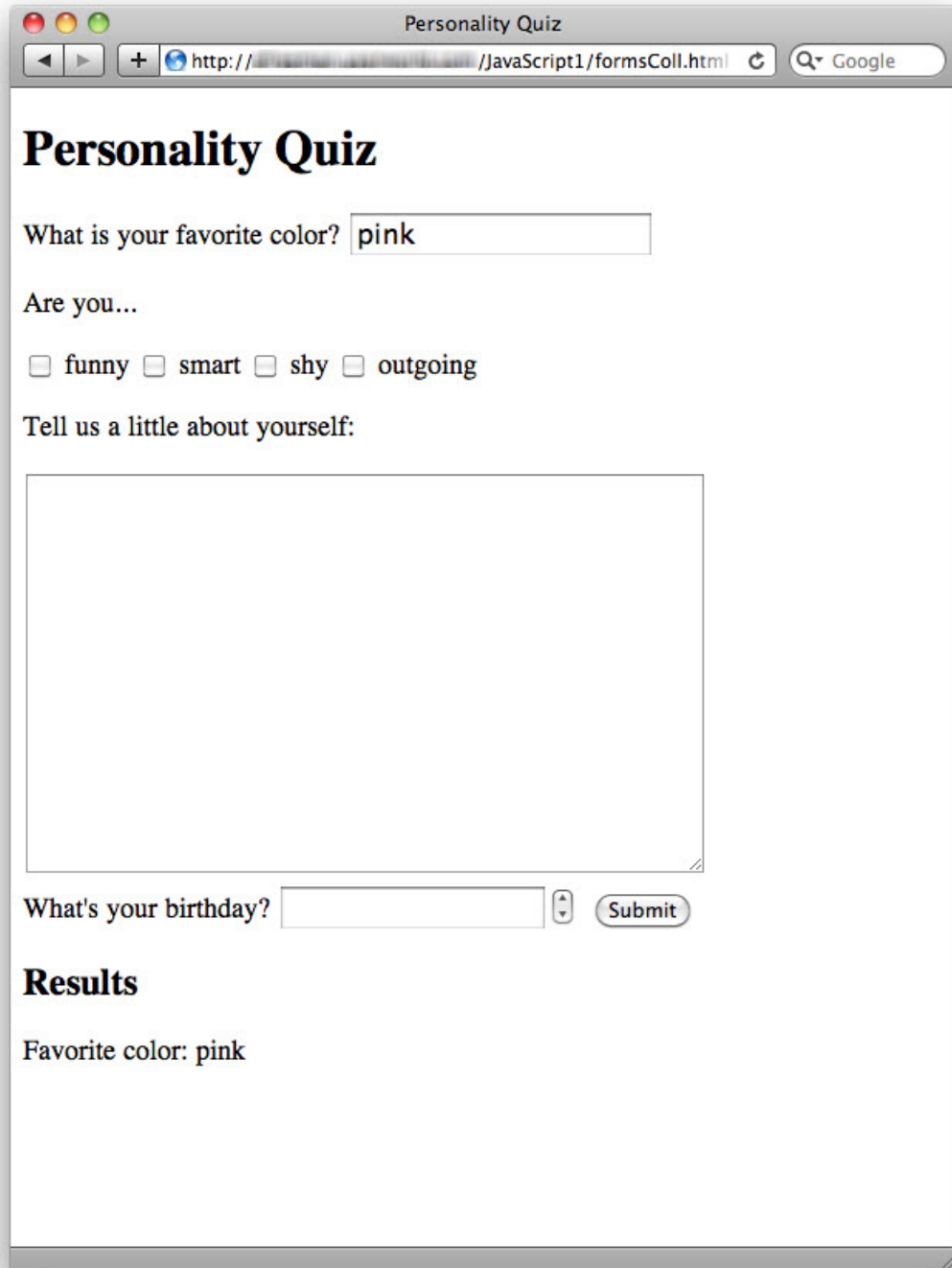
    }

    function getData() {
        var theForm = document.forms.theForm;
        var color = theForm.elements.color.value;

        var results = document.getElementById("results");
        results.innerHTML = "Favorite color: " + color;
    }
</script>
</head>
<body>
    <h1>Personality Quiz</h1>
    <form id="theForm">
        <p>
            <label for="color">What is your favorite color?</label>
            <input type="text" id="color" placeholder="blue" required>
        </p>
        <p>Are you...</p>
        <input type="checkbox" id="funny" name="personality" value="funny">
            <label for="funny">funny</label>
        <input type="checkbox" id="smart" name="personality" value="smart">
            <label for="smart">smart</label>
        <input type="checkbox" id="shy" name="personality" value="shy">
            <label for="shy">shy</label>
        <input type="checkbox" id="outgoing" name="personality" value="outgoing">
            <label for="outgoing">outgoing</label>
        <br>
        <p>Tell us a little about yourself:</p>
        <textarea id="aboutme" name="textarea" rows="12" cols="38">
        </textarea>
        <br>
        <label for="date">What's your birthday?</label>
        <input id="date" name="date" type="date">
        <input id="submitButton" type="button" value="Submit">
    </form>
    <h2>Results</h2>
    <div id="results">
    </div>
</body>
</html>

```

Save it in your work folder as *personality.html*, and open it in a browser. Enter a color and click Submit. Your favorite color appears below Results at the bottom of the page:



Just like before, when we used `document.getElementById` to get the form, we use the id of the form "theForm" to access the form. However, using the *forms collection*, we use the forms object, which is a property of the document object. As you know, to access a property of an object, you use dot notation. So `document.forms` is an object containing all the forms in the page, and that object contains a property, `theForm`, which is the form with the id "theForm," which is the form we want.

OBSERVE:

```
var theForm = document.forms.theForm;
```

The form itself is an element object. One of its properties is another collection object, the *elements collection*. This is an object that contains all the elements contained in the form. It is similar to the forms collection we just looked at. So you can use it to access elements in the form, like this:

OBSERVE:

```
var color = theForm.elements.color.value;
```

To access the elements collection, we again use dot notation. And just like before, because each form control is a property of the elements object, we can use dot notation to access a control using its id. So, to access the control with the id "color," we write `theForm.elements.color`.

We combined two steps here: getting access to the control and getting access to the value. This is perfectly okay as long as you're sure that the form control exists. If it doesn't exist, you'll get an error because the value of `theForm.elements.color` will be null! In this case, however, we're sure that the color control exists, so we won't have that problem.

Note If you need a refresher on element objects and the document object model, review Skills Ladder lesson on the Document Object Model (DOM).

Let's get the "About Me" and "Birthday" parts of the form using the forms and elements collections too:

CODE TO TYPE:

```
<!doctype html>
<html lang="en">
<head>
  <title> Personality Quiz </title>
  <meta charset="utf-8">
  <script>
    window.onload = init;

    function init() {
      var submitButton = document.getElementById("submitButton");
      submitButton.onclick = getData;
    }
  </script>
</head>
</html>
```

```

function getData() {
    var theForm = document.forms.theForm;
    var color = theForm.elements.color.value;
    var aboutme = theForm.elements.aboutme.value;
    var birthday = theForm.elements.date.value;

    var results = document.getElementById("results");
    results.innerHTML = "Favorite color: " + color + "  
";
    results.innerHTML += "About me: " + aboutme + "  
";
    results.innerHTML += "Birthday: " + birthday + "  
";
}
</script>
</head>
<body>
<h1>Personality Quiz</h1>
<form id="theForm">
<p>
<label for="color">What is your favorite color?</label>
<input type="text" id="color" placeholder="blue" required>
</p>
<p>Are you...</p>
<input type="checkbox" id="funny" name="personality" value="funny">
<label for="funny">funny</label>
<input type="checkbox" id="smart" name="personality" value="smart">
<label for="smart">smart</label>
<input type="checkbox" id="shy" name="personality" value="shy">
<label for="shy">shy</label>
<input type="checkbox" id="outgoing" name="personality" value="outgoing">
<label for="outgoing">outgoing</label>
<br>
<p>Tell us a little about yourself:</p>
<textarea id="aboutme" name="textarea" rows="12" cols="38">
</textarea>
<br>
<label for="date">What's your birthday?</label>
<input id="date" name="date" type="date">
<input id="submitButton" type="button" value="Submit">
</form>
<h2>Results</h2>
<div id="results">
</div>
</body>
</html>

```

Save it and open it in a browser. You should get the same results as before.

Using the Elements Collection to Access the Value of a Selected Radio Button

The elements collection is very handy when you want to get the value of a radio button or checkbox from a form. Let's use the forms collection to get the personality items from the form.

CODE TO TYPE:

```
<!doctype html>
<html lang="en">
<head>
  <title> Personality Quiz </title>
  <meta charset="utf-8">
  <script>
    window.onload = init;

    function init() {
      var submitButton = document.getElementById("submitButton");
      submitButton.onclick = getData;
    }

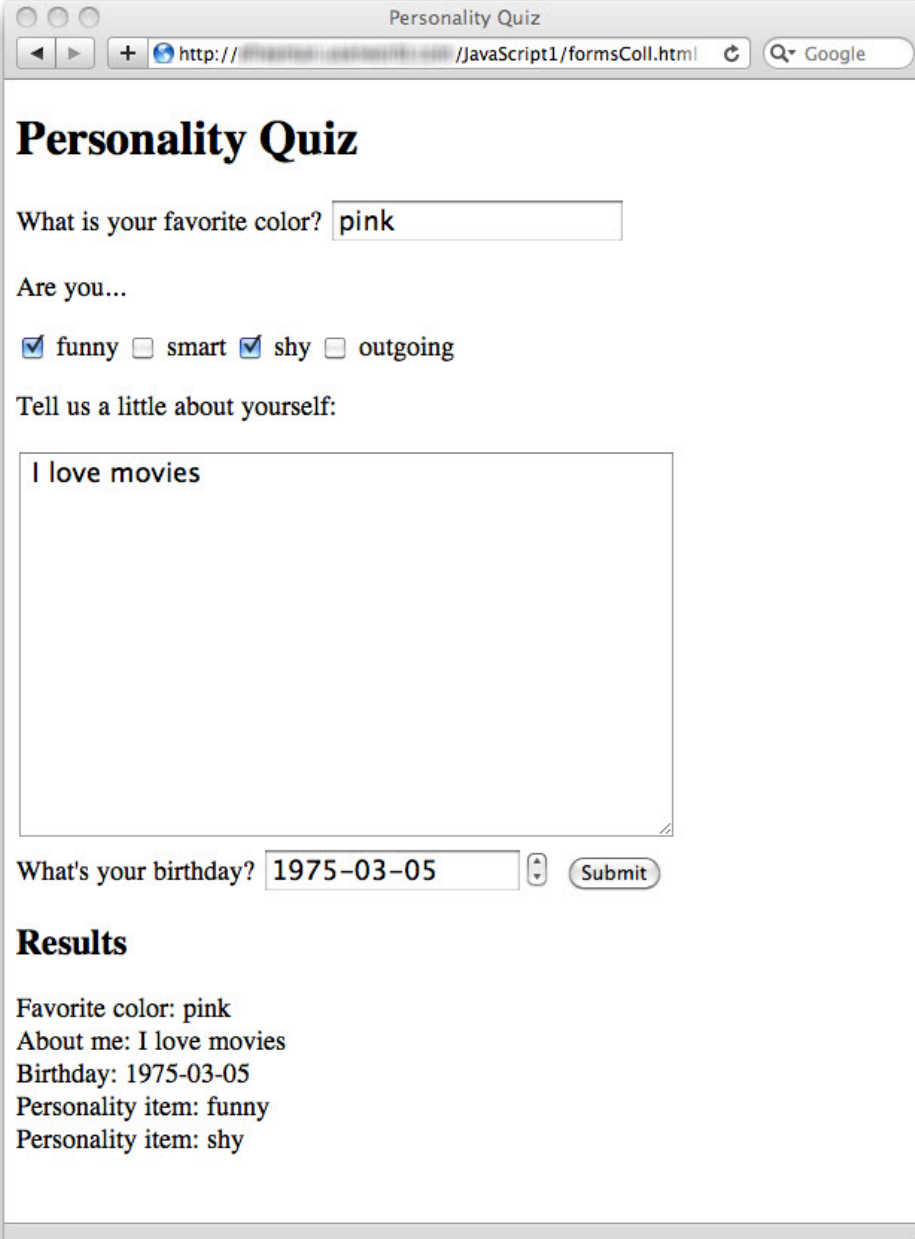
    function getData() {
      var theForm = document.forms.theForm;
      var color = theForm.elements.color.value;
      var aboutme = theForm.elements.aboutme.value;
      var birthday = theForm.elements.date.value;

      var results = document.getElementById("results");
      results.innerHTML = "Favorite color: " + color + "<br>";
      results.innerHTML += "About me: " + aboutme + "<br>";
      results.innerHTML += "Birthday: " + birthday + "<br>";

      var personalityArray = theForm.elements.personality;
      for (var i = 0; i < personalityArray.length; i++) {
        if (personalityArray[i].checked) {
          results.innerHTML += "Personality item: " +
personalityArray[i].value + "<br>";
        }
      }
    }
  </script>
</head>
<body>
  <h1>Personality Quiz</h1>
  <form id="theForm">
    <p>
      <label for="color">What is your favorite color?</label>
      <input type="text" id="color" placeholder="blue" required>
    </p>
    <p>Are you...</p>
    <input type="checkbox" id="funny" name="personality" value="funny">
      <label for="funny">funny</label>
    <input type="checkbox" id="smart" name="personality" value="smart">
      <label for="smart">smart</label>
    <input type="checkbox" id="shy" name="personality" value="shy">
      <label for="shy">shy</label>
```

```
<input type="checkbox" id="outgoing" name="personality" value="outgoing">
  <label for="outgoing">outgoing</label>
<br>
<p>Tell us a little about yourself:</p>
<textarea id="aboutme" name="textarea" rows="12" cols="38">
</textarea>
<br>
<label for="date">What's your birthday?</label>
<input id="date" name="date" type="date">
<input id="submitButton" type="button" value="Submit">
</form>
<h2>Results</h2>
<div id="results">
</div>
</body>
</html>
```

Save it and open it in a browser. Try checking one or more of the "Are you..." boxes. When you submit the form, the items you checked appear under Results at the bottom of the page:



The screenshot shows a web browser window titled "Personality Quiz". The address bar contains the URL "http://.../JavaScript1/formsColl.html". The page content includes a title "Personality Quiz", a text input field for "What is your favorite color?" with the value "pink", a section "Are you..." with checkboxes for "funny", "smart", "shy", and "outgoing" (where "funny" and "shy" are checked), a text area for "Tell us a little about yourself:" containing "I love movies", a date input field for "What's your birthday?" with the value "1975-03-05", and a "Submit" button. Below the form, a "Results" section lists the user's input: "Favorite color: pink", "About me: I love movies", "Birthday: 1975-03-05", "Personality item: funny", and "Personality item: shy".

Let's step through how this works. First, notice that each checkbox item has a different id, but has the same name, *personality*. We don't actually use the ids in this example, but we use the name to get the checkboxes as a group. HTML knows that if the checkboxes have the same name, then they are grouped together. It's also a good reminder that ids must be unique, but that the name must be the same for all checkbox (or radio button, if you're using that) items in the same group. Also notice that each checkbox item has a different *value*. The string in the

value property is what we get when we get the value of the "checked" items. In the JavaScript code, we use the *elements collection* to get the personality checkbox group from the form:

OBSERVE:

```
var personalityArray = theForm.elements.personality;
```

In this case, the `personalityArray` is a collection of checkbox elements, all with the *name* "personality." To see which checkboxes are selected, we loop through the collection and test to see if the item is checked:

OBSERVE:

```
for (var i = 0; i < personalityArray.length; i++) {  
    if (personalityArray[i].checked) {  
        results.innerHTML += "Personality item: " + personalityArray[i].value  
+ "<br>";  
    }  
}
```

When we find one that's checked, we update the "results" `<div>` to display that checkbox's value. We get the value using the `value` property, as usual, but notice that we are getting the value of the checkbox item in the collection.

You can use a similar technique to find which radio button values are selected in a form. Remember, however, that radio buttons can only have one value selected at a time, while checkboxes can have multiple values selected.

Submitting a Form with JavaScript

Earlier, we said that we used a button control in the form, rather than a submit control, so we could process the form before submitting the form to a server-side script. And in fact, in this example, we don't submit the form at all. However, there may be times when you want to validate form data, and then submit a form to a server-side script.

If you want to be able to submit a form once you've used JavaScript to validate or process the data in the form, you'd only need to make a few changes to your code:

OBSERVE:

```
<!doctype html>
<html lang="en">
<head>
  <title> Personality Quiz </title>
  <meta charset="utf-8">
  <script>
    window.onload = init;

    function init() {
      var submitButton = document.getElementById("submitButton");
      submitButton.onclick = getData;
    }

    function getData() {
      var theForm = document.forms.theForm;
      var color = theForm.elements.color.value;
      var aboutme = theForm.elements.aboutme.value;
      var birthday = theForm.elements.date.value;

      var results = document.getElementById("results");
      results.innerHTML = "Favorite color: " + color + "<br>";
      results.innerHTML += "About me: " + aboutme + "<br>";
      results.innerHTML += "Birthday: " + birthday + "<br>";

      var personalityArray = theForm.elements.personality;
      for (var i = 0; i < personalityArray.length; i++) {
        if (personalityArray[i].checked) {
          results.innerHTML += "Personality item: " +
personalityArray[i].value + "<br>";
        }
      }

      theForm.submit();
      return false;
    }
  </script>
</head>
<body>
  <h1>Personality Quiz</h1>
  <form id="theForm" method="get"
action="http://yourserver.com/yourscript.php">
    <p>
    <label for="color">What is your favorite color?</label>
    <input type="text" id="color" placeholder="blue" required>
    </p>
    <p>Are you...</p>
    <input type="checkbox" id="funny" name="personality" value="funny">
      <label for="funny">funny</label>
    <input type="checkbox" id="smart" name="personality" value="smart">
      <label for="smart">smart</label>
  </form>
</body>
</html>
```

```
<input type="checkbox" id="shy" name="personality" value="shy">
  <label for="shy">shy</label>
<input type="checkbox" id="outgoing" name="personality" value="outgoing">
  <label for="outgoing">outgoing</label>
<br>
<p>Tell us a little about yourself:</p>
<textarea id="aboutme" name="textarea" rows="12" cols="38">
</textarea>
<br>
<label for="date">What's your birthday?</label>
<input id="date" name="date" type="date">
<input id="submitButton" type="button" value="Submit">
</form>
<h2>Results</h2>
<div id="results">
</div>
</body>
</html>
```

Submitting the form in this example doesn't make much sense, and we don't have a server side script to submit it to anyway, so just take a look at how this would work in case you need to know how to do it.

Notice that all we do is get the form, and then call the form's `submit()` method. We return `false` from the function so that the JavaScript knows there's nothing else to do after the form has been submitted. That's it!

In the next lesson, you'll learn how to create new elements to add to the DOM for your page and add data you get from a form to a web page, dynamically! See you there!

More about the material in this lesson

The Riverside JS Workshop would like to acknowledge the generosity of O'Reilly Media, Inc. for making this material available to us through the Creative Commons License. We would also like to acknowledge the great work of Elisabeth Robson who authored this content. She is one of our favorites. Elisabeth has written a number of *Head First* programming books for web developers. We recommend them highly. You can browse her work [here](#).



Copyright © 1998-2015 O'Reilly Media, Inc.

This work is licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License.

See <http://creativecommons.org/licenses/by-sa/3.0/legalcode> for more information.

The original source document has been altered. It has been edited to accommodate this format and enhance readability.