

JavaScript Workshop

Form Validation

Types of form validation

- There are three ways JavaScript can be used to validate data in a form:

Keystroke - trapping individual keystroke entered by the user

Field - trapping the field change event

Form - trapping the form submit event

- The most common is Form level validation which examines all field values in the form, then alerts the user what fields have invalid data.
- The advantage of this over the other two ways is that the user gets told all once what needs to be changed instead of interrupting them after each field is completed.

Step 1: Capture form submission

- Use the form onSubmit event handler to trap the form submission so validation can occur before the form is submitted. Setting return false in the function instead would not let the form submit. The this keyword passes a reference of the form being submitted to the function.

```
function myFunction(current_form) {  
    alert("You entered: " + current_form.text1.value);  
    return true;  
}
```

```
<form name="myForm" onSubmit="return myFunction(this)">  
    <input type = text" name = "text1">  
    <input type = "submit">  
</form>
```

Step 2: Create a wrapper function

- It is best to create separate functions for the data checks and a main function that calls each one. This main function (sometimes called a wrapper or controller function) will get executed directly from the onSubmit event handler and call the other functions to do the specific validation checking.

```
function validateForm(form) {  
  
    if (form valid) {  
        return true;  
    } else {  
        alert("message about what to correct");  
        return false;  
    }  
}
```

Step 3: Check for empty fields

- To check text fields for an empty value, you can use the length property of the string object. Values returned from form fields are string values (we'll cover string objects next meeting).

```
function isEmpty(form) {  
  
    if (form.field1.value.length == 0) {  
        message += "Field1 is required" + "\n";  
    }  
  
    if (form.field2.value.length == 0) {  
        message += "Field2 is required" + "\n";  
    }  
}
```

Step 4: Check for blank space

- One issue when checking for empty fields is that the user can circumvent the check by entering blank spaces (hitting the space bar). There is no specific JavaScript function that removes leading spaces of a string value, so most of the techniques used loop through each character using String functions:

```
function strip_spaces(field) {  
  
    while (field.substring(0,1) == " ") {  
        field = field.substring(1);  
    }  
  
    return field;  
}
```

Step 5: Check for numeric data

- If numeric data needs to be entered, the `isNaN()` function can be used to validate a string value. The JavaScript function `isNaN()` takes a string parameter and evaluates it to return `true` if it is not a number and `false` if it is.
- The only issue is that `isNaN()` evaluates empty or blank spaces as a number (returns `false`). An additional technique to check for empty and/or blank spaces would need to be added.

```
function isNumber(form) {  
  
    if (isNaN(form.field1.value)) {  
        message += "Field1 is not a number" + "\n";  
    }  
}
```

Step 6: Check select and radio buttons

- Other form fields such as select boxes and radio buttons can be checked so users have to select an option or click a choice.
- Select boxes have a `selectedIndex` property (covered last meeting) that is used to check if a selection has been made.
- Radio buttons (and check boxes) have a `checked` property that returns true if a choice has been clicked.

```
function check_data(form) {
```

```
    if (form.selectedIndex == 0) confirm_msg += "Message" + "\n";
```

```
    if (form.radio.checked) confirm_msg += "Click a choice" + "\n";
```

```
}
```

- JavaScript Workshop website:
<http://www.javascriptworkshop.com>
- Any questions?
frank@javascriptworkshop.com - Frank Stepanski
- Recommended reading:
[JavaScript: A Beginner's Guide, Second Edition](#)