

Input Validation

At the end of this chapter, you will be able to:

- Use jQuery Validator library
- Use Notification dialog box
- Validate an empty text field
- Validate a text field that should contain a number with decimal(s)
- Validate a text field that should contain a whole number
- Validate a number within a range
- Round numbers to 1 or more decimal places

Instruction 1: Use jQuery Validator library

1. Form validation is a technique used to check the user's entry to ensure that only valid inputs are accepted and processed. When the user enters an invalid input, it is important to prompt him to re-enter his inputs instead of processing it.
2. Create a new project, C8.
3. Follow Instruction1 in Chapter 4 to include jQuery and jQuery Mobile libraries into your project.
4. Draw the following user interface:

5. We will write a mobile Salary Calculator with the following business logic:
 - a. Total Salary = Monthly Rate X Months Worked.
 - b. Employees with Disciplinary Action taken against them will get a 20% pay deduction.
 - c. Managers get a 25.4% bonus on top of their regular pay.
 - d. Directors get a 47.8% bonus on top of their regular pay.
 - e. Executives do not get a bonus on top of their regular pay.
6. Edit index.html:

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <!--
5       Customize the content security policy in the meta tag below as
        needed. Add 'unsafe-inline' to default-src to enable inline JavaScript.

```

```

6      For details, see http://go.microsoft.com/fwlink/?LinkID=617521
7      -->
8      <meta http-equiv="Content-Security-Policy" content="default-src
'self' data: gap: https://ssl.gstatic.com 'unsafe-eval'; style-src
'self' 'unsafe-inline'; media-src *">
9
10     <meta http-equiv="content-type" content="text/html; charset=UTF-
8" />
11     <meta name="format-detection" content="telephone=no">
12     <meta name="msapplication-tap-highlight" content="no">
13     <meta name="viewport" content="user-scalable=no, initial-
scale=1, maximum-scale=1, minimum-scale=1, width=device-width">
14     <link rel="stylesheet" type="text/css" href="css/index.css">
15     <link rel="stylesheet" href="css/jquery.mobile-1.4.5.css">
16
17     <script src="lib/jquery-1.11.2.min.js"></script>
18     <script src="lib/jquery.mobile-1.4.5.min.js"></script>
19     <script src="scripts/common.js"></script>
20
21     <title>C8</title>
22 </head>
23 <body>
24
25     <div data-role="main" class="ui-content">
26         <form name="salaryform" id="salaryform">
27             <div class="ui-field-contain">
28
29                 <div data-role="fieldcontainer">
30                     <label for="txtName">Employee Name:</label>
31                     <input type="text" name="txtName" id="txtName"
value="John Lim" required>
32                 </div>
33
34                 <div data-role="fieldcontainer">
35                     <label for="txtMonths">Months Worked:</label>
36                     <input type="number" name="txtMonths"
id="txtMonths" value="3" required>
37                 </div>
38
39                 <div data-role="fieldcontainer">
40                     <label for="txtRate">Monthly Rate:</label>
41                     <input type="number" name="txtRate" id="txtRate"
value="4505" required>
42                 </div>
43
44                 <div data-role="fieldcontainer">
45                     <label for="role">Role</label>
46                     <select name="selRole" id="selRole">
47                         <option value="1">Executive</option>
48                         <option value="1.254"
selected>Manager</option>
49                         <option value="1.478">Director</option>
50                     </select>
51                 </div>
52
53                 <div data-role="fieldcontainer">

```

```

54         <fieldset data-role="controlgroup" data-
type="horizontal">
55             <legend>Disciplinary Action Taken?</legend>
56             <label for="Yes">Yes</label>
57             <input type="radio" name="rdoDiscipline"
id="Yes" value="TRUE">
58
59             <label for="No">No</label>
60             <input type="radio" name="rdoDiscipline"
id="No" value="FALSE" checked>
61         </fieldset>
62     </div>
63
64     <input type="button" value="Calculate"
id="btnCalculate">
65 </div>
66 </form>
67 </div>
68
69     <script type="text/javascript" src="cordova.js"></script>
70     <script type="text/javascript"
src="scripts/platformOverrides.js"></script>
71     <script type="text/javascript" src="scripts/index.js"></script>
72 </body>
73 </html>

```

- a. Line 31: A “value” property has been added to the Employee Name text field. We set the value to “John Lim”. When the program runs, the default value in the Employee Name text field is “John Lim”. The “required” keyword makes this field a required entry. We will validate it to ensure that the user enters something here.
- b. Lines 36 and 41: The Months Worked and Monthly Rate text fields have also been set to their respective default values using the value properties. The “required” keyword is also used here. The type has been set to “number” thus only keys in the numeric keypad consisting of numbers, decimals and mathematical operators can be used.
- c. Lines 44 to 51: The “Manager” item in the select menu has been set as a default using the “selected” property of select menu options.
- d. Lines 53 to 62: The “FALSE” radio button in the radio options has been set as a default using the “checked” property radio button options.

7. Edit index.js:

```

1  (function () {
2
3      $(document).ready(function () {
4          $("#btnCalculate").bind("click", function () {
5              calculateSalary();
6          });
7      });
8
9      function calculateSalary() {
10         var employeename, months, rate, role, salary;

```

```

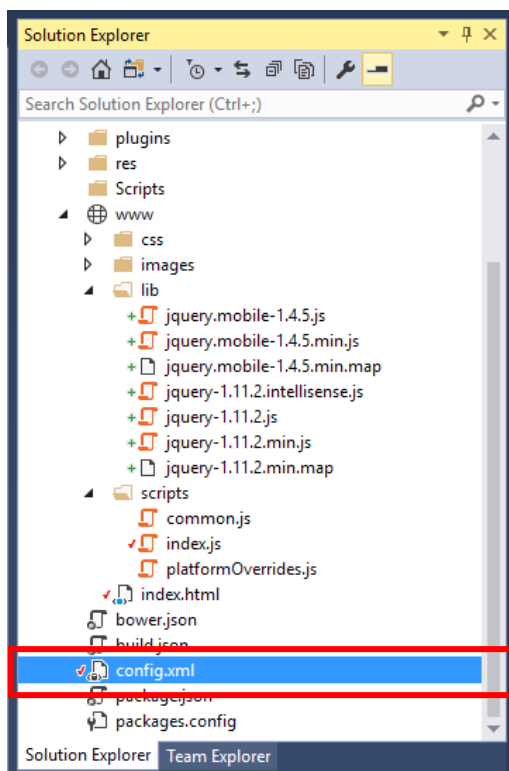
11     employeeName = $("#txtName").val();
12     months = $("#txtMonths").val();
13     rate = $("#txtRate").val();
14     role = $("#selRole").val();
15
16     alert(employeeName);
17     alert(months);
18     alert(rate);
19     alert(role);
20 }
21
22 }) ();

```

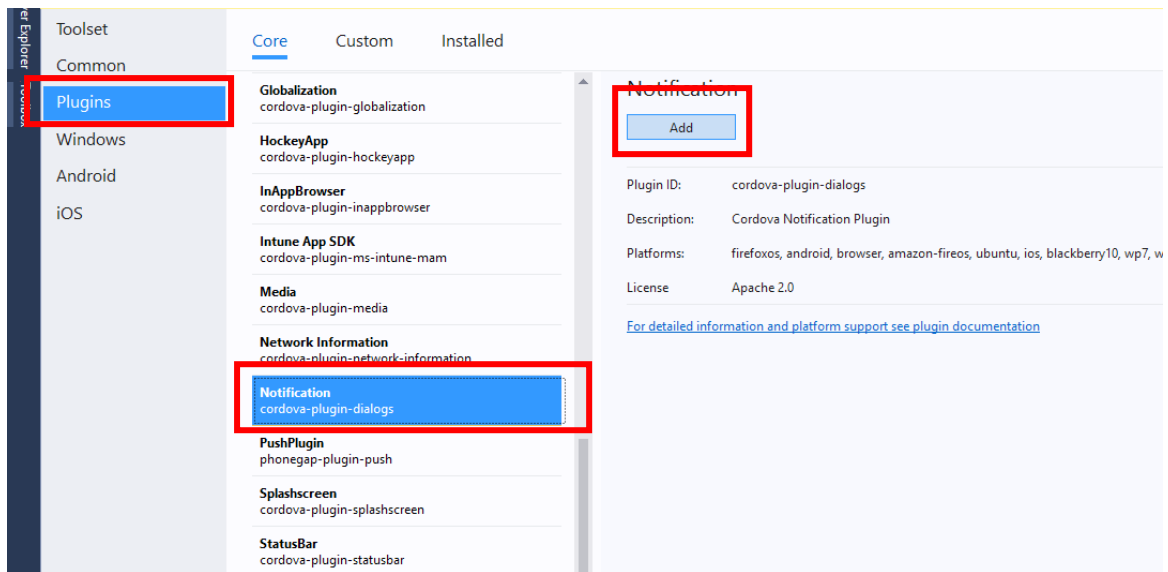
Run the program and note that the Employee Name, Months Worked, Salary Rate and Roles pop up in alert boxes.

Instruction 2: Use Notification dialog box

1. We have been using the alert() function to pop messages. The alert() function is a JavaScript function. This is not the best method to pop alerts in our mobile application. You would have noticed that in a real mobile device like an Android phone, alerts do not look like the alerts that we have been popping up.
2. To use the same kind of notifications, we will need to use a different method to pop them. First, we will need to include the Cordova Notification Plugin in our project.
3. Open config.xml in Visual Studio's Solution Explorer:



4. Select Plugins, Notification and click the [Add] button. You have now added the notification plugin and may not use its features in your project.



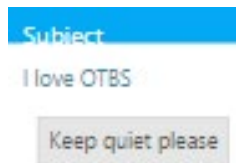
5. Add the following codes to common.js

```

1 function validationMsgs(message, title, button){
2     navigator.notification.alert(
3         message,
4         function() {},
5         title,
6         button
7     );
8 }

```

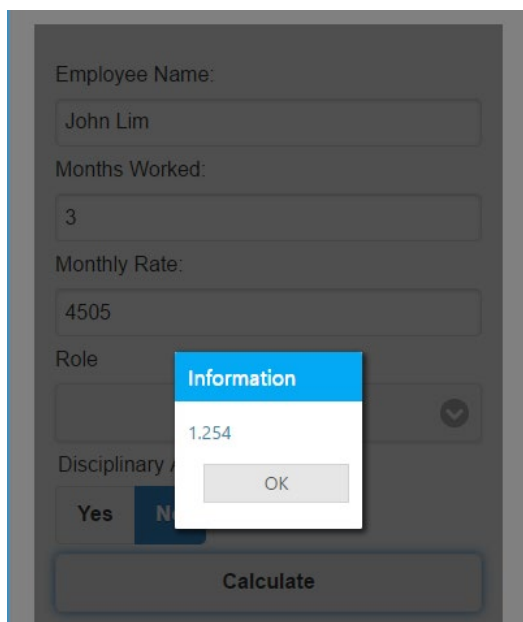
- a. Line 1 declares the function. This function is called validationMsgs() and it accepts 3 parameters, the message to display, the title of the notification pop-up, and the text to display on the button that the user will press to dismiss the function.
- b. Line 2 makes a call to the notification plugin to pop this alert.
- c. Line 3 passes the message to the notification plugin.
- d. Line 4 allows us to declare the function that will execute after the user presses the button. Currently we do not want this button to do anything, thus an empty function is placed there.
- e. Line 5 passes the title of the pop-up to the notification plugin.
- f. Line 6 passes the text that will be displayed on the button to the notification plugin.
- g. Thus, to call the function to display the following notification pop-up:



Call the function like this:

```
validationMsgs("I love OTBS", "Subject", "Keep quiet please");
```

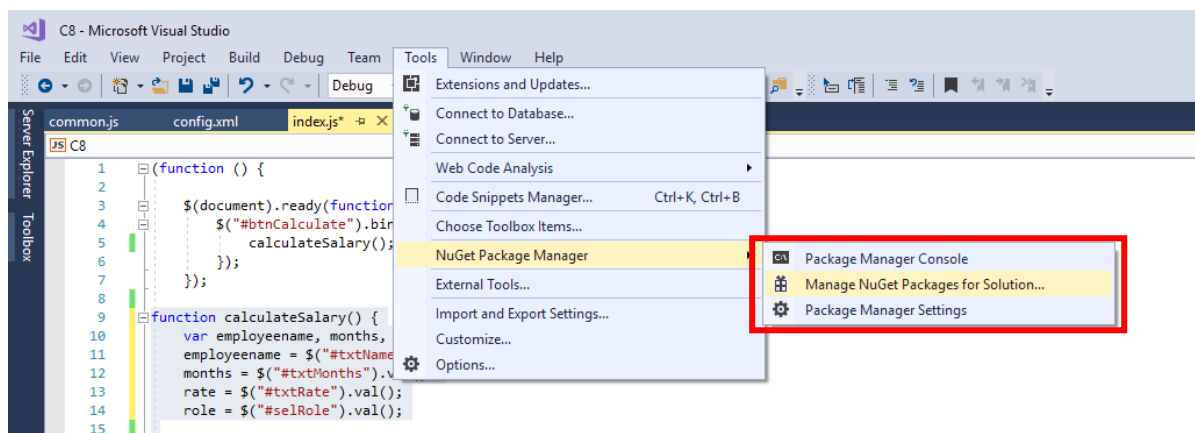
6. Edit index.js to change the alerts() to validationMsgs(). Run the app. Notice that the alerts now look like this:



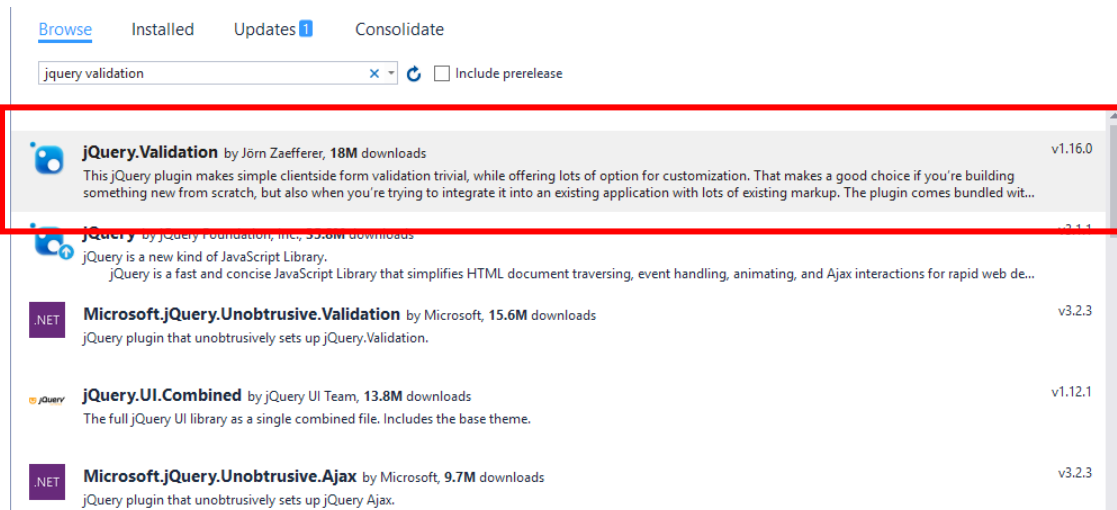
```
1 function calculateSalary() {
2     var employeename, months, rate, role, salary;
3     employeename = $("#txtName").val();
4     months = $("#txtMonths").val();
5     rate = $("#txtRate").val();
6     role = $("#selRole").val();
7
8     validationMsgs(employeename, "Information", "OK");
9     validationMsgs(months, "Information", "OK");
10    validationMsgs(rate, "Information", "OK");
11    validationMsgs(role, "Information", "OK");
12 }
```

Instruction 3: Validate an empty text field

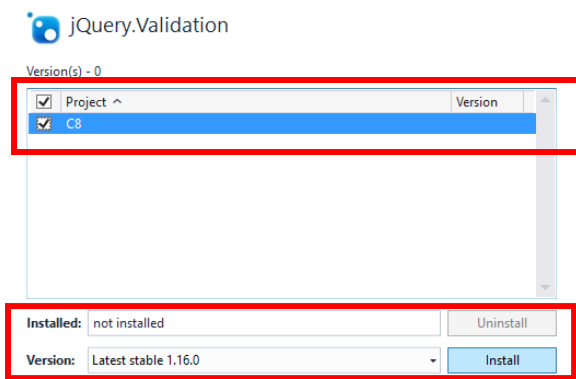
1. A common input validation is to check for a blank input when an input is required. In this example, we require the user to enter the Employee's Name. We will like to check for this input and prompt the user to enter it if the input is missing.
2. We will use a 3rd party JavaScript library to perform this validation. This library is called jQuery Validation.
3. To install this jQuery Validation library, in Visual Studio, go to Tools > NuGet Package Manager > Manage NuGet Packages for Solution...



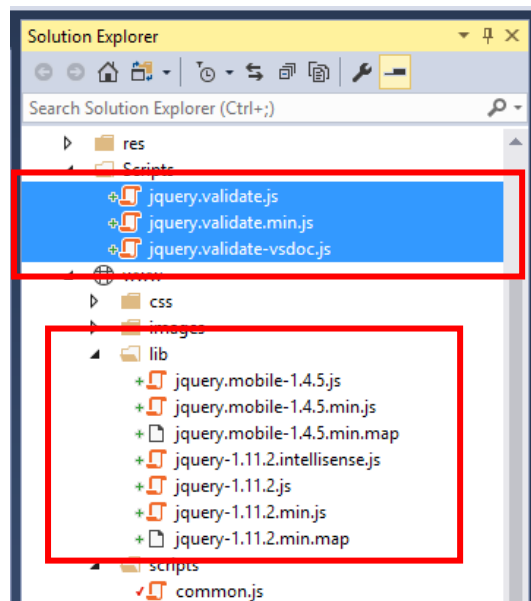
4. Search for “jQuery Validation”.



5. Click [Install] to install this library.



- The library files have been copied to the Scripts folder in Solution Explorer. Drag and drop the files to your /www/lib/ folder.



- Include the jquery.validate.min.js library in index.html:

```

1 <script src="lib/jquery-1.11.2.min.js"></script>
2 <script src="lib/jquery.mobile-1.4.5.min.js"></script>
3 <script src="scripts/common.js"></script>
4 <script src="lib/jquery.validate.min.js"></script>

```

- We now configure our validation so that the app complains if the txtName textfield which was stated as “required” has not been filled by the user. Edit index.js:

```

1 $(document).ready(function () {
2     $("#salaryform").validate({
3         messages: {
4             txtName: "employee name is required",
5         },
6         focusInvalid: false,
7         submitHandler: function () {
8             return false;
9         },
10        errorPlacement: function (error, element) {

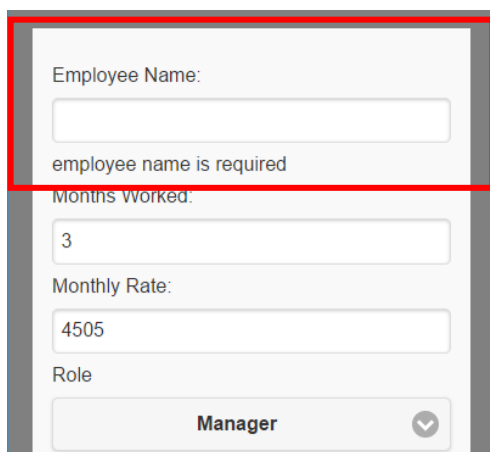
```

```

11      error.appendTo(element.parent().parent().after());
12      },
13      });
14
15      $("#btnCalculate").bind("click", function () {
16          if ($("#salaryform").valid()) {
17              calculateSalary();
18          }
19      });
20 });

```

- a. Line 2 and 13: Declares the function that validates the form. All form validations (there may be more than 1 form in an app) that we are validating will be configured within these lines. The ID and name of the form that we are validating is “salaryform”. This declares the function that validates this form.
 - b. Line 3 and 5: Messages that we will be displaying to the user will be configured within these lines.
 - c. Line 4: The message that will be displayed if txtName is invalid.
 - d. Line 6: Focuses the cursor on the last form element that is found to have an error if set to TRUE. It is currently set to false.
 - e. Line 8: Here, we can declare what we want to do after the form has been successfully submit. We currently do nothing, thus it only says “return false”.
 - f. Line 10 to 12: These code places the error message right below the text fields where the errors are.
 - g. Line 16 and 18: We run the function calculateSalary() only if the user’s entries in salaryform is valid. The function calculateSalary() does not execute if any fields are not validated correctly.
9. Run the app. Leave the “Employee Name” field blank and press [Calculate]. Note the error message appearing just below the “Employee Name” field. You must fill the text field before you can submit the form.



The screenshot shows a web form with the following fields and values:

- Employee Name:** A text input field that is currently empty. Below it, the error message "employee name is required" is displayed in red text.
- Months Worked:** A text input field containing the value "3".
- Monthly Rate:** A text input field containing the value "4505".
- Role:** A dropdown menu with "Manager" selected.

A red rectangular box highlights the "Employee Name" field and the error message below it.

Instruction 4: Validate a text file that should contain a number with decimal(s)

1. We will now configure the validation script to check if txtRate is a number. Edit index.js:

```

1 $("#salaryform").validate({
2   messages: {
3     txtName: "employee name is required",
4     txtRate: "rate must be a number",
5   },
6   focusInvalid: false,
7   submitHandler: function () {
8     return false;
9   },
10  errorPlacement: function (error, element) {
11    error.appendTo(element.parent().parent().after());
12  },
13 });

```

2. Run the app. Enter a series of “-” at txtRate. Note that the error message configured in point 1 above shows up right below the txtRate field.

Instruction 5: Validate a text field that should contain a whole number

1. We will now configure the validation script to check if txtMonths is a whole number. Edit app.js:

```

1 $("#salaryform").validate({
2   messages: {
3     txtName: "employee name is required",
4     txtRate: "rate must be a number",
5     txtMonths: "month must be a whole number",
6   },
7   rules: {
8     txtMonths: {
9       digits: true
10    }

```

```

11     },
12     focusInvalid: false,
13     submitHandler: function () {
14         return false;
15     },
16     errorPlacement: function (error, element) {
17         error.appendTo(element.parent().parent().after());
18     },
19 });

```

- a. Lines 7 and 11: These are where rules for validation are stated.
 - b. Lines 8 and 10: Rules for the txtMonths field are stated here.
 - c. Line 9: We state that only digits are allowed for txtMonths.
 - d. Line 5: We configure the message to show up if the user does not conform to the validation rule(s) for txtMonths.
2. Run the app. Enter a number with decimal at txtMonths. Note that the error message configured in point 1 above shows up right below the txtMonths field.

Instruction 6: Validate a number within a range

1. Another common validation is to check if a number entered is within a range.
2. We will validate to ensure that txtRate must be greater than 0. Edit app.js:

```

1 $("#salaryform").validate({
2     messages: {
3         txtName: "employee name is required",
4         txtRate: "rate must be a number greater than 0",
5         txtMonths: "month must be a whole number",
6     },

```

```

7   rules: {
8       txtMonths: {
9           digits: true
10      },
11      txtRate: {
12          min: 1
13      },
14  },
15  focusInvalid: false,
16  submitHandler: function () {
17      return false;
18  },
19  errorPlacement: function (error, element) {
20      error.appendTo(element.parent().parent().after());
21  },
22 });
23 });

```

- a. Lines 11 and 13 declares the validation rules for txtRate.
 - b. Line 12 validates to ensure that the minimum that the user can enter is 1.
 - c. Line 4 is updated to tell the user that he needs to ensure that rate must be a number greater than 0.
 - d. Line 10: Since there are 2 rules now, add a comma (,) after the first set of rules for txtMonths.
3. Run the app in simulate mode. Enter a negative number at at txtRate. Note that the error message configured in point 1 above shows up right below the txtRate field.

Employee Name:
John Lim

Months Worked:
3

Monthly Rate:
-3
rate must be a number greater than 0

Role
Manager

4. There are many more rules that you can configure for a form. For a complete list of validation features available as part of the jQuery Validation library, refer to <https://jqueryvalidation.org/documentation/>.

Instruction 7: Round Numbers to 1 or more decimal places

1. Rounding numbers to 1 or more decimal places is another function that is useful and have been written and shared widely. We will adopt the solution found in <http://stackoverflow.com/questions/11832914/round-to-at-most-2-decimal-places-in-javascript> in response to a question that a programmer had asked on stackoverflow.com.
2. Copy this function into common.js. This is a useful function that may be used in the programs that you may write later.

```
1 function round(number, decimals) {
2     return +(Math.round(number + "e+" + decimals) + "e-" + decimals);
3 }
```

3. To call the round function, we provide 2 arguments; the number to round, and the decimal place to round it to. For example:

```
round(3.145,2) //rounds 3.145 to 2 decimal place i.e. 3.15
```

```
round(3.145,1) //rounds 3.145 to 1 decimal place i.e. 3.1
```

4. We will complete the calculateSalary() function to use the round() function that we have just written. We will use round() to round salary to 1 decimal place as follows:

```
1 function calculateSalary(){
2     var employeeName, months, rate, role, salary;
3     employeeName = $("#txtName").val();
4     months = $("#txtMonths").val();
5     rate = $("#txtRate").val();
6     role = $("#selRole").val();
7
8     salary = round(months*rate*role, 1);
9
10    if (getRadioValue("rdoDiscipline") == "TRUE"){
11        salary = round(salary * 0.8, 1);
12    }
13
14    validationMsgs("Your Salary is " + salary, "Information", "OK");
15 }
```

- a. Line 8: The round() function is called. The result of months x rate x role is rounded to 1 decimal place.

- b. Line 10: Use the `getRadioValue()` function in `common.js` to check if the user selected “yes” or “no” for disciplinary action.
 - c. Line 11: recalculates the salary by taking 80% of its value because the employee has had his pay deducted due to disciplinary action. The `round()` function is called again in line 11. The result of `salary x 0.8` is rounded to 1 decimal place.
 - d. Line 14: Pops salary in a notification pop-up using the `validationMsgs()` function in `common.js`.
5. Run the app. Enter various values and see how the salary recalculates:

The screenshot shows a web application interface for calculating salary. It includes input fields for 'Employee Name' (John Lim), 'Months Worked' (3), and 'Monthly Rate' (4505). There is a 'Role' dropdown menu and a 'Disciplinary' section with 'Yes' and 'No' radio buttons. A 'Calculate' button is at the bottom. A blue 'Information' pop-up is displayed, showing 'Your Salary is 16947.8' and an 'OK' button.

Exercise

There are no exercises for this chapter.

Codes

Codes for this chapter can be found by searching VC8-Input-Validation on Github.com.